COMPUTATIONAL APPROACHES FOR DIFFUSIVE LIGHT TRANSPORT: FINITE-ELEMENTS, GRID ADAPTION, AND ERROR ESTIMATION

DISSERTATION

Presented in Partial Fulfillment of the Requirements for

the Degree Doctor of Philosophy in the

Graduate School of The Ohio State University

By

Richard P. Sharp Jr., B.S., M.S.

* * * * *

The Ohio State University

2006

Dissertation Committee:

Raghu Machiraju, Co-Adviser

Robert Lee, Co-Adviser

Han-Wei Shen

Approved by

Co-Adviser

Co-Adviser

Graduate Program in
Computer Science and
Engineering

ABSTRACT

Subsurface light transport in highly scattering media is a problem of great interest to both computer graphics and biomedical optics researchers. Specifically, computer graphics researchers strive to develop more accurate simulations of light physics to in turn generate more realistic synthetic images. Likewise, biomedical optics researchers are concerned with accurately simulating light propagation to aid in design of equipment for diagnostic medical use.

The mathematical formulation for diffusive light transport is presented along with a derivation for both a finite difference and finite element numerical solution for two and three dimensions. Efficient implementations are proposed which use Cholesky factorization to efficiently update the light scatter calculations if the source changes. Furthermore, the use data structures are proposed to accelerate (by an order of magnitude) parts of the source calculation and finite element matrix construction proposed by current biomedical optics literature.

Furthermore, a novel technique for grid refinement for the finite element formulation using hanging nodes is presented. This technique allows for simple mesh refinement while maintaining the flux continuity on the finite element formulation. In conjunction with this technique, a per-element error estimator derived from a Green's

function is presented along with a discussion on why traditional Galerkin style *a posteriori* estimation techniques fail. These techniques can then be combined to drive an adaptive finite element grid refinement method.

Finally, to demonstrate the practicality of these techniques are demonstrated on simulations of numerical phantoms whose geometry and scattering properties were selected using segmented MRI data of human tissues. Furthermore, the results are visually comparable to images derived from a real world device which visualizes subsurface vasculature in human tissue by transilluminating tissue with near infrared light.

To Mariana.

If they ask for me,
Tell them I've traveled the Earth
To sip tea with her.

ACKNOWLEDGMENTS

Dr. Raghu Machiraju, thank you for providing the guidance and environment that I needed graduate school to be: a place to explore. Without your help I would have never known where to go or what to look for. I will never forget that even though the hypotenuse of the minivan is shorter than the width of the mattress, it can still be tied to the roof and be driven at high speeds. You are a true scholar and an even better friend.

Thanks to Dr. Robert Lee for his patience in dealing with a graduate student (me) who had forgotten nearly all of what he learned in his undergraduate electromagnetics courses. Without you the fascinating world of finite elements, Green's functions, and the power of back of the envelope calculations would still be a mystery. Throughout my time at graduate school you have been my role model for what a researcher should strive to be.

To Dr. Han-Wei Shen, thank you for introducing me to the wonder that is programmable graphics hardware. Very few people can show their work and induce the immediate response of "cool!" as you can.

To Jacob Adams for picking up the pieces where I left off and the large amount of work he completed in generating the finite element meshes found in this dissertation. I have no doubt that you will go on to do great things.

Thanks to Dr. Robert Crane for providing a much needed application for this work and for free access to his lab and imaging equipment. I won't forget your patience with a graduate student who wanted to take picture after picture through a NIR scope with a mismatched camera nor the fascinating conversations during impromptu afternoon visits.

Without the guidance of my old officemate, Benjamin Rutt, I would have never known how to make a computer do truly useful things. With the knowledge he selflessly shared with me I have saved billions of keystrokes over the years. Without it I would have left graduate school Euchre, Emacs, Gnus, Linux, and LaTeX-free.

To my friend Jason Sawin, thanks for sharing the occasional psychosis which can be graduate school and for demonstrating the "angry method" of teaching. Very few people are as dedicated as you are and even fewer are as talented. The world will be a much better place for the thousands of students' lives you will change for the better.

Thanks to the RESOLVE group, Bruce Weide, Tim Long, Wayne Heym, Paolo Bucci, and David Mathias for their mentorship and offering me the opportunity to teach the fascinating 221/222 sequence. If it weren't for RESOLVE I would have shamefully secured my degree without the ability to design good programs.

Thanks to my father for supporting me when I decided it would be a good idea to fly tiny planes over barren deserts and rocky mountains; when I thought that joining the army would be a great idea; and nurturing my natural fascination for all things scientific when I was young. I remember it all, and am a better person for it.

Finally to Mariana, without you I would certainly be much less than I am today. Thank you for not only showing me the joy of laziness, but opening my eyes to the

world. This dissertation would not be the same or mean the same without you. From Kelly's Island to Mt. Omu; wild donkeys to sleepy buffalo; and a new baby girl to growing old together. A lifetime of adventures and true companionship are ours to live together!

VITA

August 2004	M.S. Computer Science & Engineering, The Ohio State University
December 2000	B.S. Computer Engineering, University of Utah
September 2001 – present	Graduate Research/Teaching Associate, The Ohio State University
January 1996 – August 2000	Combat Engineer (E-5), U.S. Army Corps of Engineers
August 13, 1977	Born – Salt Lake City, Utah

PUBLICATIONS

Research Publications

R. Sharp and R. Machiraju Accelerating Subsurface Scattering Using Cholesky Factorization *The Visual Computer*, pages 1–9, Jan 2006.

FIELDS OF STUDY

Major Field: Computer Science and Engineering

Studies in:

Computer Graphics	Prof. Raghu Machiraju
	Prof. Han-Wei Shen
Finite Element Methods	Prof. Robert Lee

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

Table **Page**

CHAPTER 1

INTRODUCTION

"Light is the first of painters. There is no object so foul that intense light will not make it beautiful."

— *Ralph Waldo Emerson.*

1.1 Thesis and Contributions

This dissertation addresses and answers the following questions: "What is the complete distribution of light in arbitrarily shaped tissues with varying optical properties?" and "How can the error in a finite element solution of light scattering be estimated with the diffusion approximation?" Although the scattering of light is widely studied, the simulation of scattered light in non-trivial geometries and material properties is much more complex and requires further study. In short, this dissertation describes the mathematical models which describe the propagation of light, how the process can be simulated on a computer, and how the error in that simulation can be determined. This dissertation also applies these results to synthesize realistic images involving subsurface scattering on the computer as well as show how light can be used as a diagnostic tool in medical applications.

Subsurface scattering is the process which refers to light that enters a material, becomes scattered inside, and exits in a different location and different direction than

it entered. There are several mathematical formulations to describe light propagation in space. Depending on the problem studied, light can be modeled as a wave or a particle. In general, Maxwell's Equations are applicable for "small" regions where the wave behavior of light is observable, such as light transport through sub-cellular regions. The particle model is applicable at larger scales, such as the macroscopic tissue level and would be modeled with the transport equation. This dissertation focuses on a reduction of the transport equation, namely the diffusion approximation, which is extremely accurate in predicting transport in highly scattering material.

The contributions of this dissertation benefit two realms of research: computer graphics and biomedical optics. Not only does this work borrow ideas from each area and lend to the other, but also new methods are proposed which contribute to both. In the field of computer graphics, researchers are interested in developing better methods for reproducing visual light phenomenon. While many modern methods are physically based, the physics are often sacrificed for the sake of efficiency so long as the visual quality is not affected. While this approach satisfies the visual aspect of computer graphics, it may hurt it in the long term since researchers tend toward a "hacker" disposition and loose focus on the physical processes. To address this issue, this dissertation presents two physically based approaches to subsurface scattering using the diffusion approximation with two methods of computation: finite differences and finite elements. Both are shown to generate quality images and with computational efficiencies that either operate in real time, or use precomputations to accelerate the computational process.

This dissertation also makes several contributions to the biomedical optics field. Although researchers in this area have been using finite elements to simulate scattering

for some time, this dissertation proposes advanced data structures and computational enhancements which improve the efficiency of many of the calculations. Furthermore, methods for grid refinement (using hanging nodes) and error estimation (using a novel method of moments estimator) are presented. These techniques can model light at any wavelength, so long as the particle model is accurate. In fact, as a capstone, a 3D finite element implementation of near infrared (NIR) light is is compared to a real world NIR transillumination device in Chapter 8.

The content of this dissertation is written such that not only are the contributions of the author presented, but also enough mathematical background on the subject (light transport) is given for both the sake of reference and completeness. The end goal is to provide enough information such that any graduate student of computation science may re-implement the results presented here.

The rest of this dissertation is organized as follows. In Chapter 2 previous research related to this dissertation topic is presented. In Chapter 3 the mathematical and physical framework is presented to provide the basis for Chapters 4 and 5 which use the finite difference and finite element techniques to solve the distribution of light scatter in highly diffuse materials. In Chapters 7 and 6 error estimation techniques and adaptive grid refinement are presented. Finally, Chapter 8 contains a technique which uses finite differences to simulate near-infrared light diffusion in human tissue and compares the images that this technique produces with those from those real-world devices. Conclusions and continuing work are presented in Chapter 9.

1.2 Motivation

In the field of computer graphics the study of light is often focused on the simulation of visible light for realistic image synthesis. Along this line there are essentially three popular directions of research. Although the end goal is always to develop an algorithm to recreate the visual effect of some light related behavior, several approaches are often taken to meet that goal. The first is a push to understand the physics first, then model the physics as accurately as possible to create the most realistic appearance. Often optimizations are critical in these sorts of applications so that the algorithms can complete in some reasonable amount of time. Although work in this area is few and far between, it is often fundamental for the rest of the community. Some examples of this type of research includes the classic work by Kajiya [36] which is the basis of all ray-tracing algorithms in computer graphics today. This type of work also includes the in-depth studies of "velvety" light phenomenon on human skin by Koenderink and Pont [41] and the accurate model of light reflection from wood by Marschner *et al.* [46].

The second approach is similar to the first but seeks to achieve simplifications of the physics in order to trade off visual quality with rendering efficiency. Examples of this type of work include the many simplifications of photon mapping by Jensen [33], precomputed radiance transfer by Sloan *et. al* [64], and simplifications made to the scattering model (such as only considering scattering that propagates toward the eye) to aid in hardware implementations of volume rendering by Kniss *et. al* [40].

Finally, the last approach is an attempt to simulating the resulting visual phenomenon while making physically implausible simplifications to the physics. Techniques such as these are critical to real time entertainment applications where low

quality effects are acceptable so long as the frame rate is high. Interestingly, due to the increase in computational power of console and desktop devices, research in this area is focusing toward more physically based approaches. However, many of these relatively simple and procedural techniques are still prevalent in day-to-day graphical applications such as texture mapping [28], Perlin noise [53], and the polygonal approximations to ray tracing by Shirley and Tuchman [62].

It is impossible to categorize all computer graphics research into three categories, but for the sake of discussion these reasonable characterize the research that is conducted in this area today. This dissertation attempts to deal with problems in the first area. Specifically by studying the mechanism of light diffusion in organic materials and developing techniques for the solution of the equations which describe this phenomenon as well as techniques for accelerating the calculation, estimating the error, and refining the solution.

Why make more realistic graphics? In the end, why bother developing better graphics techniques? If the end result is to just make gorier "shoot em' up" games, or prettier computer generated movies, this hardly seems to be a noble task to devote ones life to. One could also argue that research into more realistic graphics techniques should be researched for it's own sake. But, this author feels that it can be, and should be, much more.

Humans are visual beings. The understanding of ourselves self is largely dependent on what we see. Images of pristine forest, a crowded polluted city, or a sunset over the ocean can invoke a variety of states of mind across many people. When we starts to investigate why the light filters through the leaves in just that way, what makes

the pollution cause the sky to turn yellow, or why the sun distorts itself as it sets, we start to not only understand the physical nature of the world around us, but how it affects our own state of mind [54]. Learning to accurately reproduce the visual world allows us to create our own realities, and explore ourselves in a way that wouldn't otherwise be possible.

How far can it go? Is it possible to artificially reproduce the visual world so accurately that even a human couldn't tell the difference? Even using the current formal model of computation,[1] this author believes the answer is yes. Consider the following "back of the envelope" calculation.

Although there are many variables which must be factored into the design of a visual virtual reality system, two main components are arguably one of more critical aspects of such as system: resolution and computation.

Consider resolution: the human eye has a field of view of approximately $130^\circ \times 160^\circ$ and an angular resolution of about 0.02° – 0.03° [15]. Discretizing this resolution into "pixels" discretizes the resolution of the human eye into approximately 6500×8000 pixels (about 40 times larger than the author's current desktop). Due to the phenomenon of "persistence of vision" (the ability of the brain to retain the image from the human eye for a brief moment) we would need to refresh each pixel of a rate of about 60 frames per second (fps)[2]. Which brings us to the computational needs.

The next question is, how much computation would have to go into one pixel in order to render that pixel in such a way that it was indistinguishable from observing a

[1] A Turing machine.

[2] The standard video record rate for the U.S. and Japan, is 59.97, which has always looked good to this author.

"pixel" from the real world. Given that techniques exist today which can render very realistic images of complex phenomenon it is the author's opinion that approximately 1T FLOPS (10^{12} floating point operations per second) would be sufficient computing power to render a pixel with such accuracy that it was indistinguishable from reality.

So, how much power would it take to render a 6500×8000 image at 60 fps given 1T FLOPS per pixel? As a comparison the machine which the author is using to write this dissertation[3] would require about 10 minutes to render a 1T FLOPS pixel. Rendering a 1280×1024 image (the current resolution of the author's desktop) would require about 3 months, or about the same amount of time required to typeset a Ph.D. dissertation. Rendering at human eye level resolution would require just under a millennium. Certainly the author's computer is not up to the task of simulating a real-time visual environment at human level resolution, but what about in the future?

As of the time of this writing the fastest computer in the world could process 2.3×10^{15} FLOPS. This is only enough to generate a real time visual simulation for a resolution of about 86×60 at 60 fps; about 10^5 times too slow for human eye resolution. Although a speedup of 10^5 for today's supercomputer seems prohibitive, consider that the fastest computer today is 10^6 times faster than its counterpart only 9 years ago! It is conceivable that within the next 25 years such a machine would be widely available to simulate the visual physical phenomenon that we as humans consider to be reality. In the next half century the next generation console, "Nintendo True Reality" system could be much more than just a marketing gimmick.

It is the hope of this author that the work presented within will contribute to some small degree to make such a reality come true.

[3] A measly Pentium 4 2.40 GHz CPU.

CHAPTER 2

BACKGROUND AND RELATED WORK

"When a man tells you that he got rich
through hard work, ask him: 'Whose?' "

— *Don Marquis.*

While the work presented in this dissertation covers the simulation of the propagation of light in organic materials, the applications of such work are widespread. In Section 2.1, similar work in the field of computer graphics is presented to give the reader a background in the kinds of applications and previous techniques where such light studies are important. Furthermore, a large portion of this dissertation also covers the solution of the diffusion equation through the finite element method, and seeks to improve the solution through error estimation and mesh refinement techniques. The material presented in Section 2.2 describes the mathematical and physics-related work in which this dissertation is based from.

2.1 Visual Precision

In 1950 S. Chandrasekhar published *Radiative Transfer* [11] in which he presented the transport equation;[4] a description of "the propagation of electromagnetic radiation through an atmosphere which is itself emitting radiation, absorbing radiation

[4]Depending on the subject area, the transport equation is also often referred to as the "equation of radiative transfer" or the "equation of transport."

and scattering radiation." Essentially this equation treats light as the flow of "flux" and accounts for all the scattering, absorption, and generation of such flux in any medium. It is today the basis for all light propagation models in computer graphics.

The transport equation itself does not inspire any obvious algorithm for image synthesis. In fact, it wasn't until 1984 when Kajiya proposed the Rendering Equation [36] as a method for capturing all light interactions in a scene which could be captured with a synthetic camera. Essentially, Kajiya's approach was a Monte Carlo[5] ray tracing solution to the transport equation developed by Chandrasekhar. Raytracing has become so common that it is often one of the first techniques taught to computer graphics students to generate realistic images.

Although Kajiya's Rendering Equation can theoretically capture all interactions of light in a material, the cost to calculate such interactions can exponentiate in terms of the number of the number of rays traced, reflections captured, and rays spawned. Although ray tracing is commonly used today, often it is supplemented with other techniques for calculating expensive effects such as soft shadows, global illumination, and the major subject of this dissertation, subsurface scattering.

There are a few historical techniques which are not often in use today, but are still significant to the are which are worth presenting. One of the first approximations to subsurface scattering was created by Blinn in order to visually recreate the effects

[5]The Monte Carlo method, presented in [50], is a class of techniques which are used to estimate the solution of complex integrals which are otherwise too complex to solve analytically or using simpler numerical techniques. Essentially the method involves stochastically sampling the domain of a function until the method converges on a solution. Due to its slow $O(\sqrt{n})$ convergence rate (n is the number of samples) the Monte Carlo method should be avoided unless there is no other practical method for solving the problem.

Monte Carlo is also a tourist resort in the principality of Monaco, whose total area is 3/4 of a square mile, which has been famous for its gambling casinos since 1861. The citizens of Monaco, called Monégasques, are exempt from taxes, and forbidden to enter the gaming rooms. Presumably they not restricted in using Monte Carlo methods for recreational mathematics.

caused by the rings of Saturn [7]. Specifically, effects observed by the Voyager 1 and 2 probes saw that although light reflected off the rings, some light scattered through to the far side. Knowing that Saturn's rings mostly consist of dust and ice, Blinn developed an analytic single-scattering model (to be described in detail in Chapter 3) based on the angle between the viewer and the light source with respect to the orientation of the rings. This type of a model generates high quality visual results for thin uniformly lit surfaces which conform to the single scattering model.

Another expensive light calculation is that of diffuse intereflections, specifically the phenomenon that occurs when indirect light is reflected off one surface onto another. A classic example of this is the "Cornell Box" first presented in [24]. The box is a simple scene consisting of an overhead light source in a box with one open wall which the camera looks in on. The two opposing walls on the left and right are colored while the rest of the surfaces in the box are white or cream colored. When the light is switched on, light reflections from the colored walls cause the neutral walls to become colored as well. Physically, white light is hitting the colored walls, being absorbed then reflected as colored light. This colored light then hits the neutrally colored wall and reflects back into the camera, thus giving the neutral wall a colored appearance. Although this effect can be simulated with ray tracing, it requires an enormous number of rays to be traced in order to reduce the noise to some acceptable level. Using radiosity, one simplifies the problem by discretizing the surfaces and calculating the interactions in between. Although this technique is currently less popular, it is similar to the method of finite elements which is described in Chapter 5.

The most modern and widely used technique to capture global light intereflections (to include subsurface scattering, indirect lighting, and other light effects) is photon

mapping, a technique developed by H. W. Jensen [33]. Although photon mapping is still a Monte Carlo ray based approach to solving the transport equation, it makes clever use of $k-d$ trees trees [22] to reduce the number of rays necessary to generate a low noise image. Photon mapping tends to start to stray from the physics of the original transport problem by estimating power density based on local photon density; specifically this approximation breaks down in corners and edges of surfaces. The approach presented in this dissertation takes a more mathematical approach to a similar problem which has some advantages over pure photon mapping, such as increased computational efficiency and effective error estimation techniques.

Finally, since the advent of programmable graphics hardware there has been a push to move many of the expensive light calculations to the graphics processing unit (GPU). This has inspired a variety of research ranging from clever direct implementations of numerical techniques on the GPU to surprising new algorithms specifically designed for GPU hardware [10, 14, 27, 43, 48, 49, 57]. Many of these algorithms are "hacks" of otherwise physically based visual phenomenon. These techniques are only mentioned in passing since this dissertation does not deal with any GPU implementations of light scattering algorithms.

2.2 Mathematical Precision

The previous section discussed related work in the computer graphics field, that is research which was geared toward realistic image synthesis. The work presented in this section focuses on research which relates to either the mathematical modeling of light, or current computational approaches to model it.

An excellent reference for the physical properties of scattering light and the mathematics behind it is S. Prahl's 1988 Ph.D. thesis [56] in which he presents several techniques for modeling light scatter. Several of the techniques he presents such as the adding-doubling method and the Delta-Eddington approximation are only valid for one dimensional, or infinite slab configurations. However, his description for Monte Carlo light transport is excellent and is valid for any type of material in any configuration. Many researchers still use this type of solution even today.

Recently the boom in the bioengineering sector has stimulated an interest in studying light propagation in human tissue for a completely different reason. An interesting property of near infrared[6] light is that it is extremely permeable to most human tissues and can thus be used as a non-invasive diagnostic tool. Such uses include blood oxygen availability [4], brain activity monitoring [71], and near-infrared transillumination for the noninvasive observation of blood vasculature which will be presented in Chapter 8.

Researchers in biomedical optics tend to take a different approach to modeling light propagation than the computer graphics community does. While the computer graphics researchers are interested in only the visual end result thus simplifications in the physics, so long as they result in good looking results, are acceptable. On the other hand, while the biomedical community may rely on some approximations, in general the goal is to achieve as accurate computations as possible. Thus, the research in this area has a more rigorous flavor.

Biomedical simulations of light transport have traditionally been simulated with Monte Carlo techniques (as Prahl demonstrates in [56]). However, over the past

[6]Near infrared light wavelengths lie in the range of 0.75-1.4 μm.

decade, much research has been focused in using the finite element method in simulating light transport. For example, Schweiger *et al.* use the finite element method for time domain light diffusion in [59] which presents a variety of boundary conditions and two specialized source representations in two dimensions. More recent research of light transport using such techniques has been focused on correctly modeling embedded non-diffusive regions inside of diffuse ones, such as the cerebrospinal fluid (CSF) layer in the sub-arachnoid space between the skull and the brain. Interestingly, the most modern solutions to this problem involves a radiosity-style boundary condition between the finite element mesh in the diffuse region and the non-diffusive region in the CSF layer [3, 5, 6, 16, 18, 42].

This dissertation will focus on estimating the error associated with the finite element method on diffusive regions only. Although error estimators exist for finite element methods [2, 35, 51], they are inappropriate for light diffusion equations where a source is present inside the medium (see Chapter 7). Although this section only provided a historical view of the state of the art in terms of the physical and mathematical framework for light diffusion, Chapters 3, 4, and 5 contains relevant derivations as they apply to the contributions of this dissertation.

CHAPTER 3

SCATTERING OF LIGHT

"It is, indeed an incredible fact that what the human mind, at its deepest and most profound, perceives as beautiful finds its realization in external nature... What is intelligible is also beautiful."

— *Subrahmanyan Chandrasekhar.*

The theories of the *tejas* (stream of fire atoms) [55] and the disturbance of the *plenum* [23] to explain the nature of light have been disproven. Modern physics currently understands the wave-particle duality of light as developed by Albert Einstein, Louis de Broglie, and others. In essence, depending on the scale, one can mathematically model light as an electromagnetic wave, through Maxwell's Equations, or as a stream of photons, through the transport equation.

The transport equation, introduced in the previous chapter and to be presented mathematically in Equation 3.2, represents light as a "flux," the amount of photons which flow through a unit area per unit time, and does not account for the wave behavior. However, this representation is valid for most all observable light behavior[7] and is the basis for the light model in this dissertation.

[7]The flux representation of light does not account for wave effects such as polarization, interference, Doppler effect, and so on.

Although the work in this dissertation is diffusion based, the diffusion approximation is derived from the transport equation. Thus, in Section 3.1 the transport equation is presented with enough detail to understand the and implement the rest of the dissertation which depends on knowledge of the properties and terms of the original transport equation. Furthermore, common approximations to the transport equations, as well as their derivations will be presented. These include the single scattering approximation and the more important (for this dissertation) the diffusion approximation. Finally, an overview of solutions to the transport equation and its approximations will be presented. The details of the finite difference and finite element solutions to the diffusion approximation will be presented in Chapters (4 and 5).

3.1 The Transport Equation

The intensity, I, in a random medium can be divided into two parts, the reduced incident intensity I_{ri} and the diffuse intensity I_d. Reduced incident intensity is the part of the flux that remains after scattering and absorption. We denote it by $I_{ri}(\mathbf{r}, \tilde{\mathbf{s}})$ where $\mathbf{r}$ is the point at which the flux is measured and $\tilde{\mathbf{s}}$ is the unit vector in which it is propagating. Its behavior satisfies the equation

$$\frac{dI_{ri}(\mathbf{r}, \hat{\mathbf{s}})}{ds} = -\mu_t I_{ri}(\mathbf{r}, \hat{\mathbf{s}}). \tag{3.1}$$

where μ_t is a quantity called the extinction cross section. It is the summation of the absorption cross section μ_a and the scattering cross section μ_s which are both measures of rates at which photons are absorbed and scattered in a given material. The solution to Equation 3.1 is a trivial exponential decay.

The diffuse intensity is not so easy to solve for. This is intensity at the medium which is caused by light scattering through the medium to that point and whose behavior is described the transport equation. To derive this equation one can observe the properties which cause a change in the flux in some infinitesimal volume. This dissertation makes the following common assumptions/simplifications about light behavior in a volume:

- Light is monochromatic.
- The energy of the non-absorbed photons are kept the same despite any interactions with the medium.
- No internal sources exist in the medium.

Given these assumptions there are four methods which can cause a change in the photon flux through a volume (see Figure 3.1):

1. Photons which are absorbed inside the volume will decrease the flux.
2. Photons which are scattered into the direction of interest from another direction will increase the flux.
3. Photons which are scattered away from the direction of interest will decrease the flux.
4. Photons which arrive from an external source will increase the flux.

Accounting for the assumptions above, the steady state transport equation (notation taken from Ishimaru [31]) is defined as

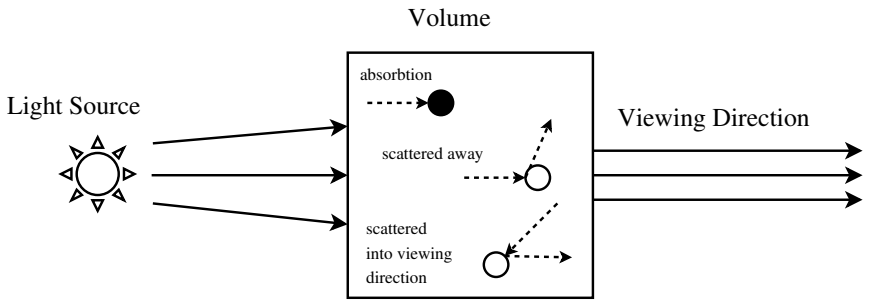


Figure 3.1: This figure illustrates the four methods of flux rate change in a volume. The incident photon stream, absorbed photons, photons scattered into the direction of interest, and photons scattered away from the direction of interest.

$$\frac{dI_d(\mathbf{r},\hat{\mathbf{s}})}{ds} = -\mu_t I_d(\mathbf{r},\hat{\mathbf{s}}) + \frac{\mu_s}{4\pi}\int_{4\pi} p(\hat{\mathbf{s}},\hat{\mathbf{s}}')I_d(\mathbf{r},\hat{\mathbf{s}}')d\omega' + \varepsilon_{ri}(\mathbf{r},\hat{\mathbf{s}}) \tag{3.2}$$

where $p(\hat{\mathbf{s}},\hat{\mathbf{s}}')$ is a phase function (described later in this section) which accounts for the distribution of scatter and $\varepsilon_{ri}(\mathbf{r},\hat{\mathbf{s}})$ is the source function due to reduced incident intensity defined below

$$\varepsilon_{ri}(\mathbf{r},\hat{\mathbf{s}}) = \frac{\mu_t}{4\pi}\int_{4\pi} p(\hat{\mathbf{s}},\hat{\mathbf{s}}')I_{ri}(\mathbf{r},\hat{\mathbf{s}})\; d\omega'. \tag{3.3}$$

The first term of the transport equation defines the flux at a point $\mathbf{r}$ and a direction $\hat{\mathbf{s}}$, the second accounts for photons absorbed in the medium, the third accounts for photons scattered into and out of the direction of interest, while the last term accounts for a source outside of the medium.

Boundary Condition The diffuse intensity is the part of the intensity which can only be generated by scattering. Since, I_d can only be generated from within the medium, there should be no inward directed diffuse intensity at the boundary. Formally stated

$$I_d(\mathbf{r}, \hat{\mathbf{s}}) = 0 \qquad \text{on the boundary when } \hat{\mathbf{s}} \cdot \hat{\mathbf{n}}^+ < 0 \tag{3.4}$$

Where $\hat{\mathbf{n}}^+$ is the outward directed normal on the surface.

Phase Functions The phase function, $p(\hat{\mathbf{s}}, \hat{\mathbf{s}}')$, in Equation 3.2 describes the probability that an incoming particle in the direction $\hat{\mathbf{s}}$ will be deflected in a direction $\hat{\mathbf{s}}'$. Essentially, this is a probability distribution of a scattered particle given the angular distance between the incoming and outgoing vector. Along with accurate scattering and absorption cross section values (μ_s and μ_a) the selection of the phase function is critical to accurately modeling the light scattering behavior of the medium.

The phase functions used in this dissertation are normalized in the same manner as by astrophysicists [11], namely the integration over all angles of the function is 1. In three dimensions this is stated mathematically as

$$\int_{4\pi} p(\hat{\mathbf{s}}, \hat{\mathbf{s}}')\, d\omega = 1. \tag{3.5}$$

The selection of a phase function depends on the sizes and shapes of the particles which scatter light in the medium. As an example, an isotropic phase function (equally likely to scatter in any direction) would be simply[8]

$$p_{\text{iso}}(\hat{\mathbf{s}}, \hat{\mathbf{s}}') = \frac{1}{4\pi}. \tag{3.6}$$

Modified HG Phase Function A common phase function to model the scattering of light in human dermis, used by Prahl [56], Jacques *et al.* [32] and Yoon *et al.* [72] is

[8]Note that the term 4π in Equation 3.6 and others occurs because of an integral of a constant over all 4π steradians and the normalization of the phase function; in two dimensions this constant would be 2π.

the Modified Henyey-Greenstein (HG) function. This function is customizable with two parameters, χ and g. The variable χ ranges from 0 to 1 and defines the amount of anisotropy present in the phase function. A value of 0 indicates fully isotropic scattering, while 1 indicates completely anisotropic behavior. The second parameter g is the average mean cosine and is defined as the integral over all angles of the phase function multiplied by the cosine of the angle between the incoming and outgoing directions

$$g = \int_{4\pi} p(\hat{\mathbf{s}}, \hat{\mathbf{s}}')(\hat{\mathbf{s}} \cdot \hat{\mathbf{s}}')\, d\omega. \tag{3.7}$$

For a normalized phase function, g ranges from -1 to 1, where -1 indicates a completely backscattering scenario (all incoming particles are scattered directly opposite from the direction of incidence), 1 is fully forward scattering, and 0 is completely isotropic. Many physical measurements of tissue scattering properties include values for g (see Cheong *et al.* [12]).

Given the understanding of these variables, the modified HG function is

$$\rho_{m-HG}(\theta) = \frac{1}{4\pi}\left[\chi + (1-\chi)\frac{1-g^2}{(1+g^2-2g\cos\theta)^{3/2}}\right]. \tag{3.8}$$

The results included in this dissertation in later chapters use the modified HG function as the phase function in all related calculations.

3.2 Approximations to the Transport Equation

Under certain physical conditions the transport equation can be simplified for easier calculation. Two of these simplifications are presented here. The single scattering approximation is useful in media when the ratio of the volume occupied by

particles to the total volume of the medium is considerably smaller than 0.1% [31]. For example, atmospheric effects such as haze can be accurately modeled using the single scattering approximation. When the volume density is much greater than 1% the diffusion approximation gives good solutions which are simpler to solve than the transport equation directly.

3.2.1 Diffusion Approximation

This section details the steps to derive the diffusion approximation from the transport equation and follows the same derivation as given in Ishimaru [31]. It is necessary to rederive this equation for the sake of exposition since parts of it are used heavily in other chapters.

The diffusion approximation is a simplification of the transport equation that expresses the distributed light intensity not as a vector, but as a scalar. The intuition being that diffusion has scattered light almost equally in all directions, thus the intensity at a point need no longer be defined in terms of the direction of observation. Mathematically we can express the diffuse incident intensity as a summation of a uniform diffuse intensity (independent of direction) plus a small propagation term[9]. Formally this is expressed as

$$I_d(\mathbf{r}, \hat{\mathbf{s}}) \simeq U_d(\mathbf{r}) + c\mathbf{F}_d(\mathbf{r}) \cdot \hat{\mathbf{s}} \tag{3.9}$$

where c is a constant and $\mathbf{F}_d(\mathbf{r})$ is the diffuse flux vector in the direction $\hat{\mathbf{s}}_\mathbf{f}$ and is defined by

[9]This propagation term is necessary to propagate the flux in the medium, otherwise the flux would be zero everywhere in the medium.

$$\mathbf{F}_d(\mathbf{r}) = \int_{4\pi} I_d(\mathbf{r}, \hat{\mathbf{s}})\hat{\mathbf{s}}\, d\omega = F_d(\mathbf{r})\hat{\mathbf{s}}_\mathbf{f}, \tag{3.10}$$

where $F_d(\mathbf{r})$ is the magnitude of the diffuse flux vector and $U_d(\mathbf{r})$ is the average diffuse intensity defined by

$$U_d(\mathbf{r}) = \frac{1}{4\pi}\int_{4\pi} I_d(\mathbf{r}, \hat{\mathbf{s}})\, d\omega. \tag{3.11}$$

To solve for the constant c, note that

$$F_d(\mathbf{r}) = \mathbf{F}_d(\mathbf{r}) \cdot \hat{\mathbf{s}}_f = \int_{4\pi} I_d(\mathbf{r}, \hat{\mathbf{s}})(\hat{\mathbf{s}} \cdot \hat{\mathbf{s}}_f)\, d\omega \tag{3.12}$$

and substitute the approximation of $I_d(\mathbf{r}, \hat{\mathbf{s}})$ from Equation 3.9 into 3.12:

$$\begin{aligned}
F_d(\mathbf{r}) &= \int_{4\pi} [U_d(\mathbf{r}) + c\mathbf{F}_d(\mathbf{r}) \cdot \hat{\mathbf{s}}](\hat{\mathbf{s}} \cdot \hat{\mathbf{s}}_f)\, d\omega && (3.13)\\
F_d(\mathbf{r}) &= \hat{\mathbf{s}}_f \cdot \int_{4\pi} [U_d(\mathbf{r}) + c\mathbf{F}_d(\mathbf{r}) \cdot \hat{\mathbf{s}}]\, \hat{\mathbf{s}}\, d\omega && (3.14)\\
F_d(\mathbf{r}) &= c\, \hat{\mathbf{s}}_f \cdot \int_{4\pi} [\mathbf{F}_d(\mathbf{r}) \cdot \hat{\mathbf{s}}]\, \hat{\mathbf{s}}\, d\omega && (3.15)\\
F_d(\mathbf{r}) &= c\, \hat{\mathbf{s}}_f \cdot \frac{4\pi}{3}\mathbf{F}_d(\mathbf{r}) && (3.16)\\
F_d(\mathbf{r}) &= c\, \frac{4\pi}{3} F_d(\mathbf{r}) && (3.17)\\
c &= \frac{3}{4\pi}. && (3.18)\\
& && (3.19)
\end{aligned}$$

Thus, the diffusion approximation of the diffuse intensity is[10]

$$I_d(\mathbf{r}, \hat{\mathbf{s}}) = U_d(\mathbf{r}) + \frac{3}{4\pi}\mathbf{F}_d(\mathbf{r}) \cdot \hat{\mathbf{s}}. \tag{3.20}$$

[10]Note that the terms of Equation 3.20 are in fact the first two terms of the Taylor series expansion of I_d about the powers of $\hat{\mathbf{s}} \cdot \hat{\mathbf{s}}_f$.

Continuing with the diffusion equation derivation, integrate Equation 3.2 over all solid angles to obtain a power relationship

$$\nabla \cdot \mathbf{F}_d(\mathbf{r}) = -4\pi\mu_a U_d(\mathbf{r}) + 4\pi\mu_s U_{ri}(\mathbf{r}) \tag{3.21}$$

where $U_{ri}(\mathbf{r})$ is the average reduced incident intensity defined by

$$U_{ri}(\mathbf{r}) = \frac{1}{4\pi}\int_{4\pi} I_{ri}(\mathbf{r},\hat{\mathbf{s}})\; d\omega. \tag{3.22}$$

Then, substituting Equation 3.20 into Equation 3.2 yields

$$\nabla U_d(\mathbf{r}) = -\frac{3}{4\pi}\mu_{tr}\mathbf{F}_d + \frac{3}{4\pi}\int_{4\pi} \varepsilon_{ri}(\mathbf{r},\hat{\mathbf{s}})\hat{\mathbf{s}}\; d\omega. \tag{3.23}$$

where μ_{tr} is the transport cross section $(\mu_s(1-g)+\mu_a)$. Next, solve for $\mathbf{F}_d$ in Equation 3.23

$$\frac{3}{4\pi}\mu_{tr}\mathbf{F}_d = -\nabla U_d(\mathbf{r}) + \frac{3}{4\pi}\int_{4\pi} \varepsilon_{ri}(\mathbf{r},\hat{\mathbf{s}})\hat{\mathbf{s}}\; d\omega \tag{3.24}$$

$$\mathbf{F}_d = -\frac{4\pi}{3}(\mu_{tr})^{-1}\nabla U_d(\mathbf{r}) + (\mu_{tr})^{-1}\int_{4\pi} \varepsilon_{ri}(\mathbf{r},\hat{\mathbf{s}})\hat{\mathbf{s}}\; d\omega \tag{3.25}$$

and then the result of Equation 3.25 can be substituted into Equation 3.21 to replace $\mathbf{F}_d$ as

$$\begin{aligned}\nabla \cdot \left(-\frac{4\pi}{3}(\mu_{tr})^{-1}\nabla U_d(\mathbf{r}) + (\mu_{tr})^{-1}\int_{4\pi} \varepsilon_{ri}(\mathbf{r},\hat{\mathbf{s}})\hat{\mathbf{s}}\; d\omega \right) = \\ -4\pi\mu_a U_d(\mathbf{r}) + 4\pi\mu_s U_{ri}(\mathbf{r}) + \int_{4\pi} \varepsilon(\mathbf{r},\hat{\mathbf{s}})\; d\omega.\end{aligned} \tag{3.26}$$

Finally, simplifying Equation 3.26 yields the diffusion approximation:

$$\nabla \cdot (\mu_{tr}^{-1})\nabla U_d(\mathbf{r}) - 3\mu_a U_d(\mathbf{r}) = -3\mu_s U_{ri}(\mathbf{r}) + \frac{3}{4\pi}\nabla \cdot \left[\mu_{tr}^{-1} \int_{4\pi} \varepsilon_{ri}(\mathbf{r}, \hat{\mathbf{s}})\hat{\mathbf{s}}\, d\omega\right] \quad (3.27)$$

which can be further simplified to

$$\nabla \cdot \beta\nabla U_d(\mathbf{r}) - \lambda U_d(\mathbf{r}) + S(\mathbf{r}) = 0 \quad (3.28)$$

where

$$\beta = (\mu_{tr})^{-1} \quad (3.29)$$

$$\lambda = 3\mu_a \quad (3.30)$$

and

$$S(\mathbf{r}) = \nu U_{ri}(\mathbf{r}) - \frac{3}{4\pi}\nabla \cdot \left[\beta \left(\int_{4\pi} \varepsilon_{ri}(\mathbf{r}, \hat{\mathbf{s}})\hat{\mathbf{s}}\, d\omega\right)\right] \quad (3.31)$$

$$= \nu U_{ri}(\mathbf{r}) - \frac{3}{4\pi}\nabla \cdot \beta\mathbf{Q}_1(\mathbf{r}) \quad (3.32)$$

where

$$\mathbf{Q_1}(\mathbf{r}) = \frac{\mu_t}{4\pi} \int_{4\pi} \left[\int_{4\pi} p(\hat{\mathbf{s}}, \hat{\mathbf{s}}')\hat{\mathbf{s}}\, d\omega\right] I_{ri}(\mathbf{r}, \hat{\mathbf{s}}')d\omega' \quad (3.33)$$

and

$$\nu = 3\mu_s. \quad (3.34)$$

Note that if the medium is isotropic (constant phase function) then the source term reduces to

$$S(\mathbf{r}) = \nu U_{ri}(\mathbf{r}). \tag{3.35}$$

Diffusion Boundary Conditions Recall that the boundary on the transport equation allowed no diffuse flux to enter the medium from the outside at the boundary. Since the diffusion approximation is expressed independently of direction an approximate boundary condition can be derived that limits the integration of the total inward diffuse flux to be zero. This can be expressed in terms of U_d only as

$$\mu_{tr} U_d(\mathbf{r}) - \frac{2}{3} \frac{\partial}{\partial n^-} U_d(\mathbf{r}) + \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi} = 0 \tag{3.36}$$

where $\partial / \partial n^-$ is the normal derivative in the direction inward from the boundary and $\hat{\mathbf{n}}^-$ is the normal pointing inward.

3.3 Solution Methods

The rest of this dissertation focuses on how to solve Equation 3.27 to accurately model light diffusion problems in highly scattering media. Specifically, Chapter 4 solves the diffusion equation using the finite difference method and demonstrates how Cholesky Factorization can be used for rendering subsurface scattering effects on the computer at interactive rates. Chapter 5 will examine how the finite element technique can be used to solve the diffusion equation with the added benefit of local error estimation techniques and mesh refinement.

CHAPTER 4

FINITE DIFFERENCE TECHNIQUES

"If God has made the world a perfect mechanism, He has at least conceded so much to our imperfect intellect that in order to predict little parts of it, we need not solve innumerable differential equations, but can use dice with fair success."

— *Max Born.*

In Chapter 3 the diffusion approximation, the simplification of the transport equation in highly scattering media, was derived. In this Chapter a solution to the diffusion equation using finite differences is derived and, among other models, it is applied to an MRI derived model of the human foot whose inhomogeneous scattering values are set based on the tissue types in the foot.

The work presented here (previously published in [60] and [61]) is not the first attempt at solving the light diffusion equation using finite differences. Jos Stam presented an implementation of multiple scattering as a diffusion processes in [65]. Stam's diffusion model was also derived from Ishimaru [31] but solved the diffusion process through a multi-grid finite difference scheme and a finite-element blob method. However, the work presented here models an interesting connection between light transport through a medium and voltage distributions on a resistive circuit grid.

Furthermore the work presented here shows how a Cholesky factorization of the resulting finite difference matrix can be used to accelerate subsurface scattering for repeated light calculations on the same geometry.

4.1 Derivation

The transport model derived here proves that the solution for light propagation through highly scattering media is analogous to voltage propagation through a resistive network. Although this process can be implemented numerically, the circuit model is presented as an interesting analogy. The derivation in this section shows how light is related to current and how the solution of a resistive network is the same as the solution to the diffusion approximation.

4.1.1 Irradiance is Related to Voltage

Irradiance (radiant power per unit area) is the integration of incoming radiance over all directions, its units are W/m^2. Hence, irradiance can be formed in terms as power per unit area:

$$\frac{d\phi}{dA} = E = \frac{p}{dA}. \tag{4.1}$$

Likewise, power can be realized as a product of voltage and current:

$$p = \frac{dw}{dt} = \frac{dw}{dq} \cdot \frac{dq}{dt} = vi. \tag{4.2}$$

From Ohm's law, Equation 4.2 can be rewritten in terms of voltage only:

$$p = \frac{v^2}{R}. \tag{4.3}$$

Resistor R has no physical significance in the solution other than to scale the initial voltage level, hence it is chosen arbitrarily.[11] Thus, the relationship between voltage and irradiance is:

$$E \propto \frac{v^2}{dA} \Rightarrow v \propto \sqrt{E \cdot dA}. \tag{4.4}$$

4.1.2 Diffusion Approximation in terms of Kirchoff Current Laws

Recall the diffusion approximation from Chapter 3

$$\nabla \cdot (\mu_{tr}^{-1}) \nabla U_d(\mathbf{r}) - 3\mu_a U_d(\mathbf{r}) = -3\mu_s U_{ri}(\mathbf{r}) + \frac{3}{4\pi} \nabla \cdot \left[\mu_{tr}^{-1} \int_{4\pi} \varepsilon_{ri}(\mathbf{r}, \hat{\mathbf{s}}) \hat{\mathbf{s}} \, d\omega \right] \tag{4.5}$$

which can be expressed in a simpler form

$$\nabla^2 u - au = \varepsilon \tag{4.6}$$

where
- u is the diffusion term
- a absorption coefficient
- ε source term.

Assume the volume of interest is discretized into equal Cartesian cubes of width h. Using first order forward and backward difference operators Equation 4.6 can be expressed in finite difference form:

$$\begin{aligned} \varepsilon_{ijk} = \quad & -au_{ijk} + \tfrac{1}{h^2} \left[\triangleright_x(u_{ijk}) + \triangleleft_x(u_{ijk}) + \triangleright_y(u_{ijk}) + \right. \\ & \left. \triangleleft_y(u_{ijk}) + \triangleright_z(u_{ijk}) + \triangleleft_z(u_{ijk}) \right] \end{aligned} \tag{4.7}$$

where the forward and backward difference operators $\triangleright_x$ and $\triangleleft_x$ are defined as

[11] The author commonly selected a value of 1kΩ.

$$\begin{aligned} \triangleright_x(u_{i,j,k}) &= u_{i+1,j,k} - u_{i,j,k} \\ \triangleleft_x(u_{i,j,k}) &= u_{i-1,j,k} - u_{i,j,k}. \end{aligned} \tag{4.8}$$

As shown previously, it is possible to represent irradiance in terms of voltage or current. Thus, $u_{i,j,k}$ is treated the voltage potential in the middle of a finite volume centered at position (i, j, k). To model the potential for current to spread to neighboring finite elements neighboring volumes are connected through discrete resistors. The values of said resistors represent the likelihood of current to pass through the space between the two points, much like a phase function.

Thus, Equation 4.7 can be expressed as a current equation in terms of node voltages and resistances for every cell (i, j, k):

$$\begin{aligned} \frac{\Delta_V V s_{i,j,k}}{R_{s_{i,j,k}}} = \; & -\frac{V_{i,j,k}}{R_{g_{i,j,k}}} + \frac{1}{h^2}\left(\frac{\Delta_V V_{i+1,j,k}}{R_{i+,j,k}} + \frac{\Delta_V V_{i-1,j,k}}{R_{i-,j,k}} + \right. \\ & \frac{\Delta_V V_{i,j+1,k}}{R_{i,j+,k}} + \frac{\Delta_V V_{i,j-1,k}}{R_{i,j-,k}} + \\ & \left. \frac{\Delta_V V_{i,j,k+1}}{R_{i,j,k+}} + \frac{\Delta_V V_{i,j,k-1}}{R_{i,j,k-}}\right) \end{aligned} \tag{4.9}$$

where $\Delta_V v \equiv v - V_{i,j,k}$ and $R_{i+,j,k}$ is defined as the resistor connecting nodes (i, j, k) and $(i + 1, j, k)$, similarly, $R_{i-,j,k}$ indicates the resistor connecting nodes (i, j, k) and $(i - 1, j, k)$ and so on. These resistor values represent preferred scattering directions in the material, where the lower the value, the higher the scatter. These resistors are generalization of the extinction and scattering coefficient.

Recall from Section 3.2.1 that the boundary condition in the diffusion equation requires that the enforcing that the inward integral of the diffuse intensity on the boundary must be zero. This condition can be enforced on the finite difference formulation by using the following equation for boundary nodes:

$$\frac{V_{i',j',k'} - V_{i,j,k}}{h} = 0 \tag{4.10}$$

where node (i', j', k') is the closest node to the boundary along the normal.

Since this model is based on a diffusion approximation note that it will not be able to handle scattering in following material types

- Transparent materials, since there is very little scattering in such materials, thus causing the transparent appearance.
- Materials too thin to allow the diffusion effect to occur.

In both these cases a first order (single scattering) approximation would be a better model to capture these effects.

Validation To validate the model a two dimensional 15×15 grid was implemented to compare with the diffusion approximation given by the Jensen *et al.* model [34]. The optical properties that were chosen were within range of the "Wholemilk" data recorded in the same paper.

The input to both grids was a step function shown in Figure 4.1(a). In the model presented in [34] this corresponds to a unit level of irradiance incident on the surface, while in the model presented here it corresponds to setting the middle group of cells' source voltage to approximately 1 volt. The results shown in Figures 4.1(b) and (c) show that the response from the [34] model and the impedance network to be quite similar.

4.2 Implementation

This section highlights the critical design choices made when implementing the scattering/impedance diffusion model presented in Section 4.1. Specifically, the storage choice, solution technique, efficiency enhancements, and rendering algorithm are presented.

4.2.1 Storage

As presented earlier, Equation 4.9 defines the scattering equation for every cell. Since this model limits light transport to only neighboring cells, each equation will be dependent on at most six other equations. Thus, the resulting matrix representing the system of equations is sparse where an impedance grid of size $i \times j \times k$ will generate a matrix of dimensions $(i \cdot j \cdot k)^2$ with at most $6i \cdot j \cdot k$ non-zero elements.

To store the linear system the author used a compressed column storage mode used by the TAUCS linear solver library [67]. This technique requires approximately $2d + 2n$ space to store n non-zero elements in a matrix of size $d \times d$.

4.2.2 Solving the system

Once the matrix representing the linear system of equations is built (using Equation 4.9 and the known scattering properties of the medium) a system of the form

$$[A]\{x\} = \{b\} \tag{4.11}$$

must be solved. The matrix $[A]$ is defined by Equation 4.9 and vector b contains the source values found at each node. In general, Equation 4.9 generates a matrix that is

both symmetric and positive definite (so long as resistor values are chosen such that the matrix remains diagonally dominant).

4.2.3 Efficiency Enhancements

Precomputation There are many methods for solving systems of the form $[A]\{x\} = \{b\}$. However if $[A]$ is positive-definite, Cholesky decomposition is an economical method for decomposing $[A]$ into the form

$$[A] = [L][L]^T. \tag{4.12}$$

Once the factor is calculated, one solves $[A]\{x\} = \{b\}$ by first solving $[L]\{y\} = \{b\}$ for $\{y\}$, then $[L]^T\{x\} = \{y\}$ for $\{x\}$. Solving these smaller problems is an inexpensive process since both vectors $\{y\}$ and $\{x\}$ can be directly solved using back and forward-substitution. Furthermore, if $[L]$ is sparse the computation time decreases further.

Unfortunately, simply because $[A]$ is sparse does not guarantee that its Cholesky factor will also be. In fact, in the worse case the factor could be a full matrix! Fortunately, graph partitioning algorithms exist which can be used to precondition a matrix such that the fill ordering of its factor is also sparse [39]. In this implementation the author used the sparse matrix package, TAUCS [67], which preconditions the matrix using the the multilevel graph partitioning algorithm implemented in the METIS [38] library.

Using these techniques, Table 4.1 shows the fill rates given original number of non-zeros in the matrix. In general, Cholesky factorization increases the fill of the factor by less than an order of magnitude. The "Upper Bound Fill" column was estimated by assuming that in the worst case, every unknown would fill an upper triangular row

completely. Note that actual fill rates are at least two orders of magnitude smaller than that of a potentially full factor.

Efficient Source Calculation The vector b defines the value of source voltages at all the nodes in the scattering mesh. The value of the voltage at a node should be proportional to that of the value of ε_{ri} at the same point. However, reasonable simplifications can be made to reduce the complexity of calculating b. First, the diffusion approximation is only valid in materials where scattering is the dominant transport mechanism. Practically, this means that the mean path, defined as the average distance a photon travels before scattering, is quite short with respect to the size of the global geometry. Thus the value of ε_{ri}, which is defined as the amount of light which is unscattered and unattenuated, drops off significantly a small distance beneath the surface where it has little effect on the final solution. One can take advantage of this exponential falloff by only calculating values for b in cells which lie on, or touch the surface of the object. Internal cells are assumed to have no source.

Finally, to determine if a cell has any light incident upon it, one can trace a shadow ray from the corners of the cell to the light source. If any of the corners of the cell are illuminated the voltage is initialized based on the inner product of the average normal of the polygons with the light vector. In order to accelerate the ray test, which is critical for interactive scattering updates, an octree data structures is used with with eight levels and up to six primitives per leaf to be stored in the leaves (see Figure 4.2).

Since the scattering matrix $[A]$ is independent of any light source, the factor $[L]$ can be kept in memory and $\{b\}$ can be updated when the light source changes. One

Al model

Grid Size	Factor Time	U_{ri} Time	Solve Time	Non-Zeros in $[A]$	Non-Zeros in $[L]$	Upper Bound Fill for $[L]$
64×64×32	$25.438s$	$7.250s$	$0.453s$	2.40×10^5	5.37×10^6	8.00×10^8
32×32×16	$1.593s$	$1.422s$	$0.046s$	3.92×10^4	3.99×10^5	2.13×10^7
16×16×8	$0.031s$	$0.281s$	$< 0.001s$	6.76×10^3	5.35×10^4	6.35×10^5
8×8×4	$< 0.001s$	$0.046s$	$< 0.001s$	1.09×10^3	4.70×10^3	1.65×10^4

Dolphin model

Grid Size	Factor Time	U_{ri} Time	Solve Time	Non-Zeros in $[A]$	Non-Zeros in $[L]$	Upper Bound Fill for $[L]$
64×64×32	$29.750s$	$2.578s$	$0.359s$	1.26×10^5	1.54×10^6	2.20×10^8
32×32×16	$0.610s$	$0.594s$	0.047	2.21×10^4	1.35×10^5	6.78×10^6
16×16×8	$0.032s$	$0.141s$	$0.016s$	4.60×10^3	1.56×10^4	2.94×10^5
8×8×4	$< 0.001s$	$0.031s$	$< 0.001s$	9.00×10^2	3.46×10^3	1.13×10^4

Table 4.1: Runtime statistics for "Al" and "Dolphin" model (See Figures 4.3 and 4.4). $[A]$ is the scattering matrix, while $[L]$ is its Cholesky factor. Note that the performance gain can be considered to be the amount of time taken to solve the entire system (the sum of "factor", "solve" and "source calculation" time) as compared to factoring once, and reusing the factor to repeatedly solve the system. There is additional run time overhead involved in copying the solution to a 3D texture in the graphics card.

can then reuse the factor to determine the new scattering distribution. Table 4.1 shows the speedup obtained by using the factor during the scattering rendering step.

4.2.4 Rendering Algorithms

To summarize, the precomputation algorithm is as follows:

1. Build global matrix $[A]$ based on Equation 4.9.
2. Compute Cholesky factorization, $[L][L]^T$.
3. Compute light source vector $\{b\}$ (using octree data structure to accelerate intersection computations).
4. Solve for $\{x\}$ using $[L]$.

5. If the location or intensity of the light source changes go to step 3.

4.3 Rendering

The general rendering algorithm is as follows:

1. Load a 3D mesh.

2. Build a scattering grid to fit inside the mesh[12].

3. Set the resistor values on the internal grid[13].

4. Perform precomputation algorithm described in Section 4.2.3.

5. Render the image by treating the solved grid as a 3D luminare and projecting it onto the surface of the 3D model[14]. The nodal voltages are proportional to the intensity values exiting the material.

6. If the location or intensity of the light source[15] changes go to step 6.

4.4 Results and Conclusions

Figures 4.3 and 4.4 show the differences between scattering and no scattering on public domain `.obj` models with constant internal impedance. Although the scattering model can be used to calculate subsurface scattering effects on inhomogeneous

[12]The mesh can be generated with many methods, but for our implementation we embedded a cube of cells around the model, then removed cells which were outside of the surface.

[13]In several examples in this paper we set the values based on a mapping from the MRI scalar data with the transfer function, see Figures 4.5 and 4.6 and [60] for details.

[14]In Figures 4.2, 4.3 and 4.4 we used OpenGL to do the rendering, while Figures 4.5 and 4.6 are rendered in Maya.

[15]There is no limitation that the light source be a point source. In fact, Figures 4.5 and 4.6 both are illuminated from area light sources.

models such as Figure 4.5 and 4.6, the author has found that the calculation of the light source vector is too expensive to calculate in real time due to the higher grid resolution required to accurately capture the self shadowing effects in the subsurface.

The time expended on precomputation and solving and non-zero fill stages are described in Table 4.1. Although denser meshes should give more accurate results, experiments show that low density meshes give appealing visual results while also being coarse enough to update the subsurface scatter at interactive rates (approximately 0.5-1 frames per second (fps) on a $16 \times 16 \times 8$ grid). Since the implementation was realized in OpenGL additional overhead was incurred with updating the 3D texture per frame as the light source moved, thus the render times are not only the source calculation plus solve times as shown in Table 4.1 but also the time necessary to copy the 3D texture to the graphics card. However, if the light source is fixed the rendering becomes trivial since the 3D texture need not be updated. In this case completely interactive (30+ fps) rates can be achieved.

4.5 Discussion

The finite difference technique can be used to simulate subsurface scattering effects in diffuse materials for visual purposes. An advantage of the finite difference method is that is is easily implemented for any problem which can be fit into a Cartesian grid. Although finite difference methods can be adapted to non-cubic grids it is difficult representation to implement. The next chapter contains a solution to the diffusion approximation using the finite element method which is easily adaptable to arbitrary geometries.

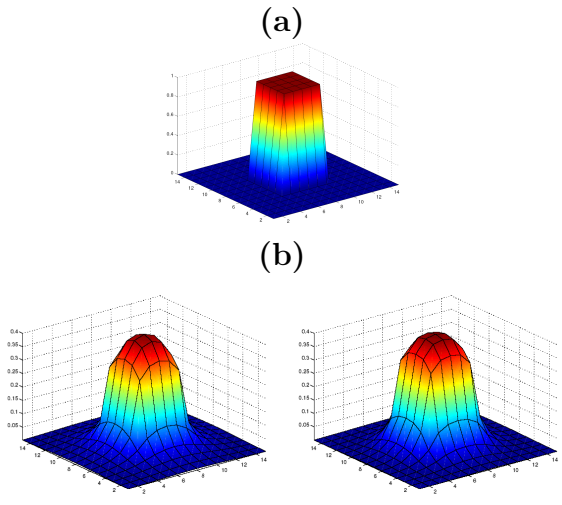


Figure 4.1: (a) The step initial source value. (b) Response of Jensen *et al.* model with $\sigma_a = 0.0041$ and $\sigma_s' = 2.6$ (left) vs. Response of our model with 1.18V step input and all resistors set to 1kΩ (right).

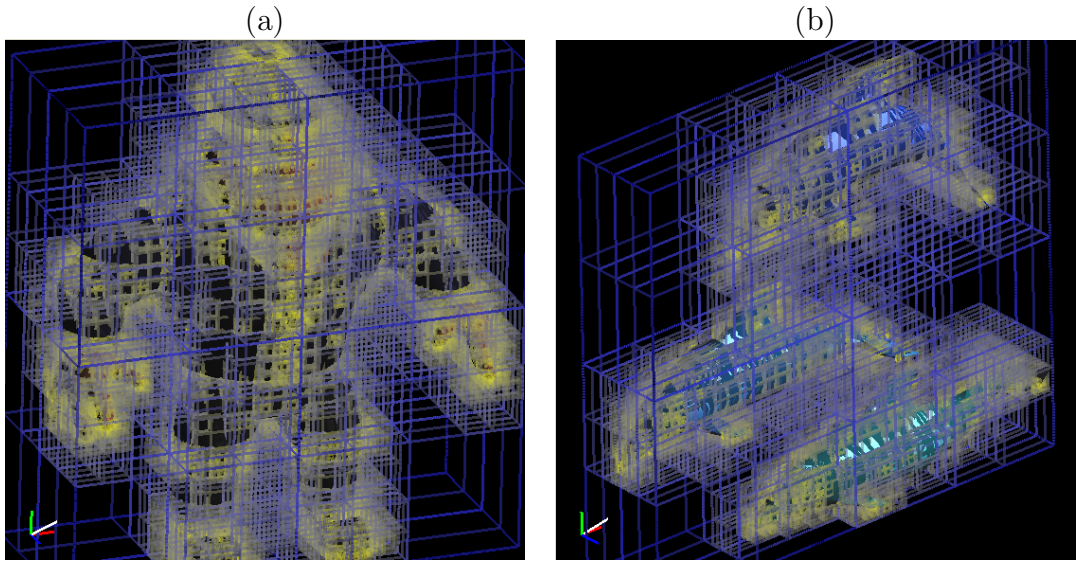


Figure 4.2: Octree used in calculating source vector for (a) "Al" model and (b) "Dolphin" model. Levels range from blue (lowest) to yellow (highest).

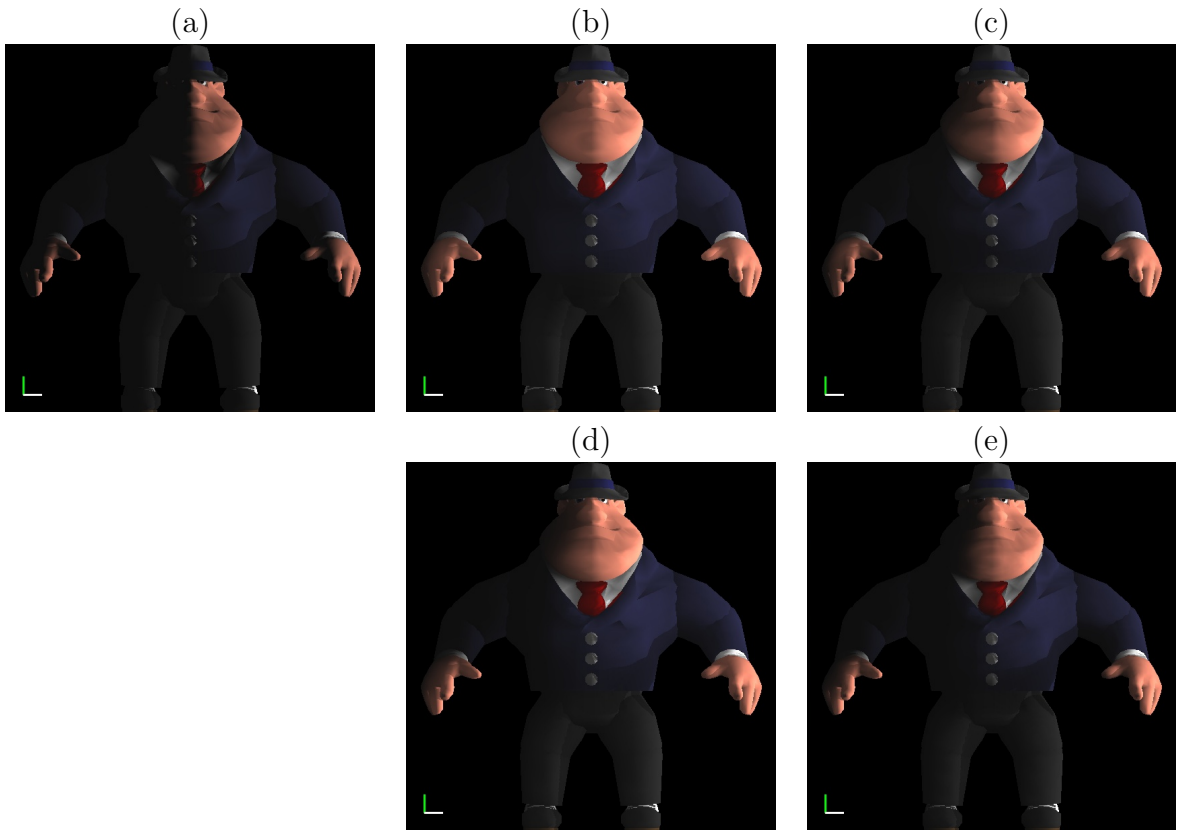


Figure 4.3: (a) "Al" model with no subsurface scattering and with embedded subsurface grids of size (b) 8×8×4, (c) 16×16×8, (d) 32×32×16 and (e) 64×64×32.

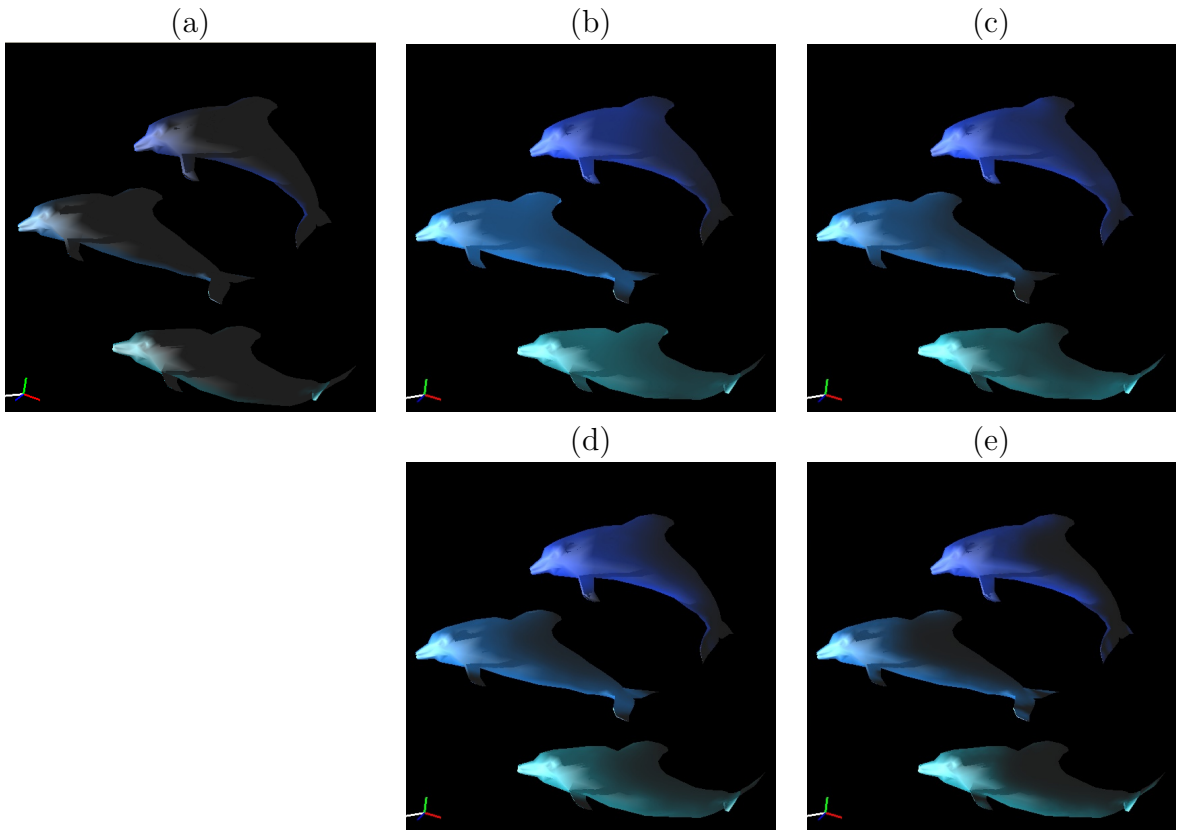


Figure 4.4: (a) "Dolphin" model with no subsurface scattering and with embedded subsurface grids of size (b) 8×8×4, (c) 16×16×8, (d) 32×32×16 and (e) 64×64×32.

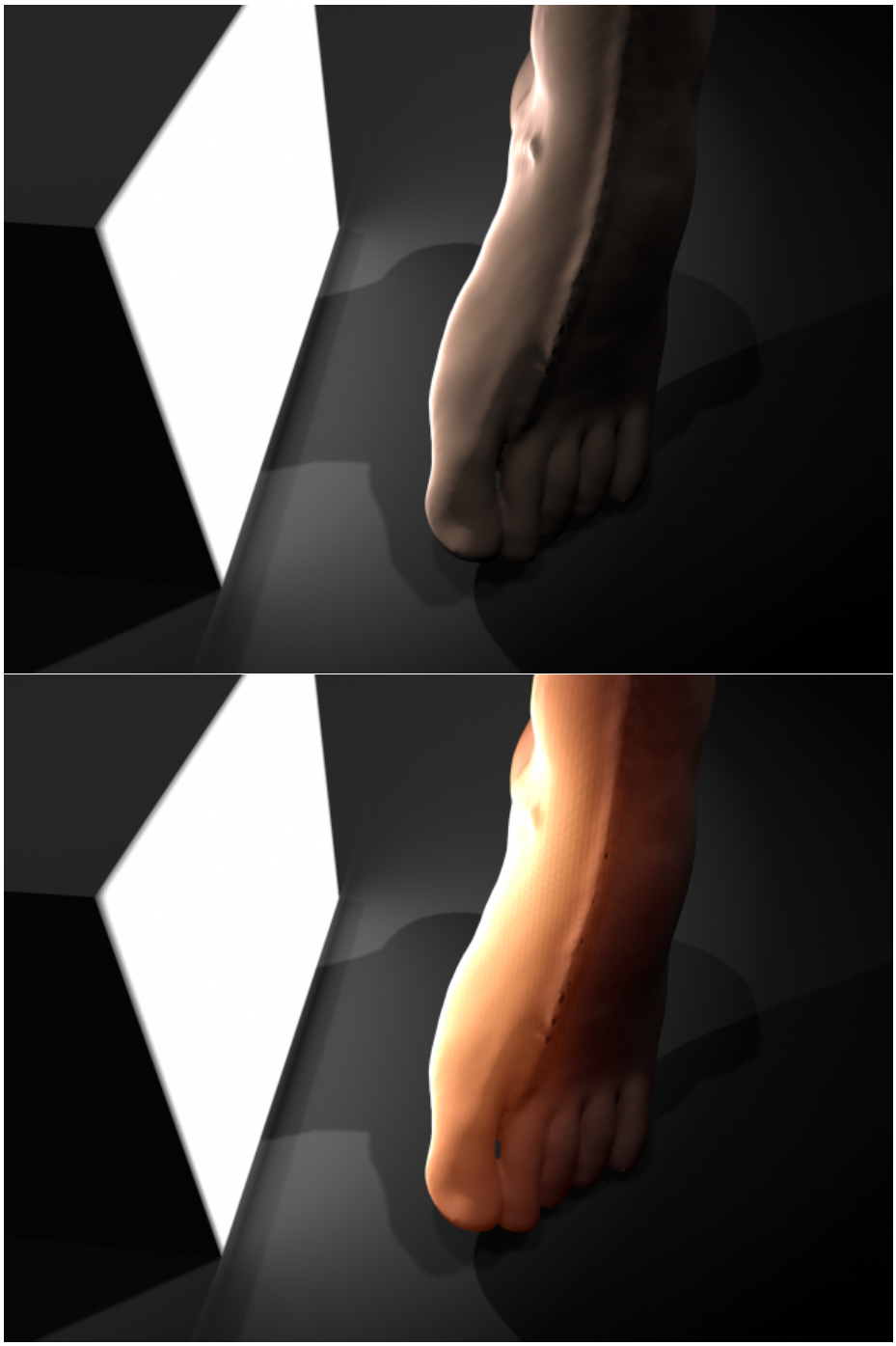


Figure 4.5: Side lit foot with no scattering vs. scattering.

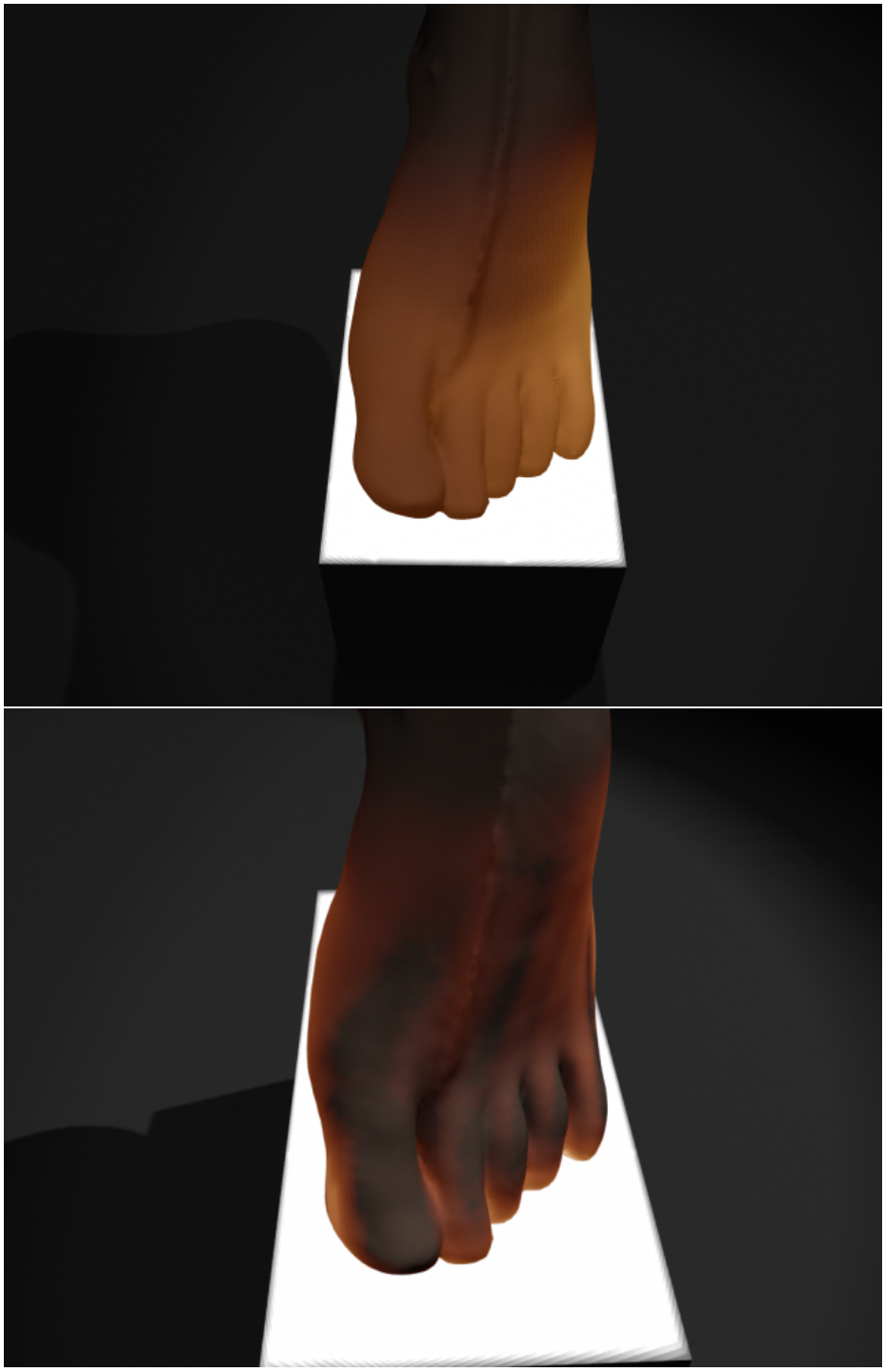


Figure 4.6: Bottom lit foot with scattering but uniform impedance vs. bottom lit foot with non-uniform impedance values based on MRI density values.

CHAPTER 5

FINITE ELEMENT APPROACHES

"Science is a differential equation.
Religion is a boundary condition."

— Alan Turing.

In the previous chapter a finite difference solution was derived for the steady state light diffusion equation and applied to real world data to generate visually realistic results. However, errors exist in the finite difference formulation whenever a Cartesian grid cannot be perfectly embedded in the geometry. In this chapter, a finite element solution to the diffusion approximation is proposed. It has the advantage of being easily applied to grids of various shapes and thus allows refinement on the grid on regions where the error is high. Later chapters will illustrate a novel grid adaptation and error estimation scheme.

5.1 General Derivation

This section will formulate the system of equations for the diffusion equation using Galerkin's method [2]. First, the formulation will be presented in a general form and in later sections refined to two and three dimensional domains. Galerkin's method is a weighted residual method which seeks the solution by weighting the residual of the

differential equation. In the case of Galerkin's method, the weighting functions are selected to be the same functions which interpolate the solution.

The finite element method solves for solutions along the nodes of the discretized domain. Given the solutions at the nodes we can derive an expression for the unknown solution in an element using the interpolation functions

$$U_d^e(\mathbf{r}) = \sum_{j=1}^{n} \phi_j^e(\mathbf{r}) U_{d_j}^e = \{\phi^e(\mathbf{r})\}^{\mathrm{T}} \{U_d^e\} = \{U_d^e\}^{\mathrm{T}} \{\phi^e(\mathbf{r})\} \tag{5.1}$$

where n is the number of nodes in the element, $U_{d_j}^e$ is the value of U_d at node j of the element e, ϕ_j^e is the interpolation function for node j of element e and the notation $\{\star\}$ defines a column vector of the form $[\star_1 \ \star_2 \ \ldots \ \star_n]^{\mathrm{T}}$.

5.1.1 Domain Discretization (2D)

In this dissertation, the two dimensional finite element grids will use triangular elements over the domain Ω with the following common restrictions: elements are connected via their vertices and no vertex of one element can be internal to the side of another. Furthermore, one should avoid generating narrow triangular elements or elements having a small inner angle since they tend to increase the solution error.[16]

The elements and nodes are labeled with separate sets of integers for identification (element numbers and node numbers). Furthermore it is convenient for formulating the system of equations to have the concept of "local node numbers" and "global node numbers." The local node number is simply the number assigned to a vertex within an element, in the case of triangular elements, element number e will have local nodes

[16]The reason that "poorly" shaped triangles cause greater error occurs because the element itself is still linearly interpolated. Thus, the longer the element the less accurate the interpolation scheme becomes.

1, 2, 3. While the global numbers of each node will likely be different. A connectivity array is a conceptual device used to relate local node number, global node number and element number to each other. In the two dimensional problem here this array has dimensions $3 \times M$ where M is the number of elements. The connectivity array is denoted $n(i, e)$ where i is the local node number, e is the element number and the value is the global node number.

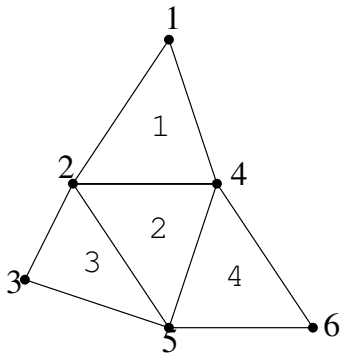


Figure 5.1: Example 2D subdivision

5.1.2 Domain Discretization (3D)

In three dimensions the volume is subdivided into a number of small volume elements, the derivation for this section will subdivide using tetrahedrons. Here, element e will have local node numbers $i = 1, 2, 3, 4$. An example element is shown in Figure 5.2.

For this dissertation the faces bounding this figure are arbitrarily defined as follows:

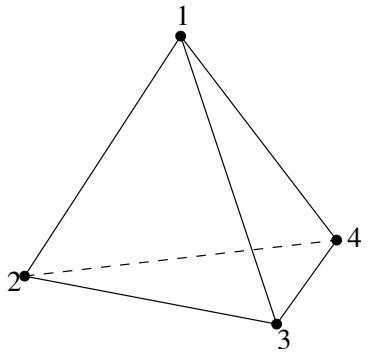


Figure 5.2: Linear tetrahedral element.

Face	Vertex 1	Vertex 2	Vertex 3
1	1	2	3
2	1	3	4
3	1	4	2
4	4	3	2

5.1.3 2D Elemental Interpolation

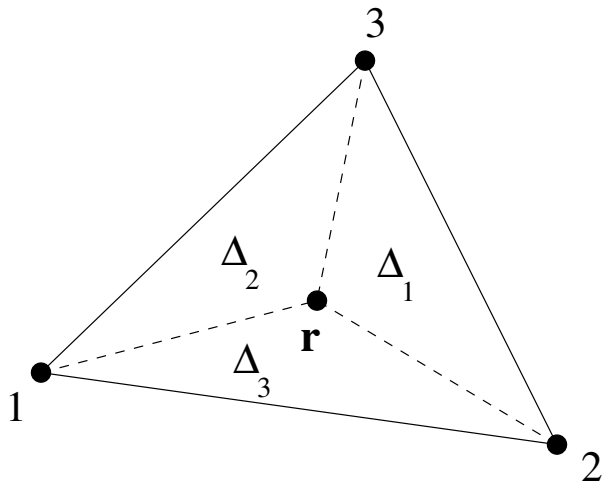


Figure 5.3: Point **r** within a triangular element.

In this section the linear interpolation functions ϕ are defined. Noting Figure 5.3 the area of the triangle given by the points $\mathbf{r}23 = \Delta_1$ is defined as

$$\begin{aligned} \Delta_1 &= \frac{1}{2}\begin{vmatrix} 1 & x_r & y_r \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} && (5.2) \\ &= \frac{1}{2}[(x_2y_3 - x_3y_2) + r_x(y_2 - y_3) + r_y(x_3 - x_2)]. && (5.3) \end{aligned}$$

where (x_r, y_r) is the xy coordinate of the point of interpolation and $(x_2, y_2), (x_3, y_3)$ are the xy coordinates for nodes 2 and 3 respectively. Furthermore, the following useful constants are defined

$$\begin{array}{lll} a_1^e = x_2^e y_3^e - y_2^e x_3^e; & b_1^e = y_2^e - y_3^e; & c_1^e = x_3^e - x_2^e \\ a_2^e = x_3^e y_1^e - y_3^e x_1^e; & b_2^e = y_3^e - y_1^e; & c_2^e = x_1^e - x_3^e \\ a_3^e = x_1^e y_2^e - y_1^e x_2^e; & b_3^e = y_1^e - y_2^e; & c_3^e = x_2^e - x_1^e. \end{array} \quad (5.4)$$

Using the notation defined in Equation 5.4, Equation 5.3 and the equations for the other two sub-triangles can be rewritten as as functions of $\mathbf{r}$

$$\begin{aligned} \Delta_1(\mathbf{r}) &= \frac{1}{2}(a_1^e + r_x b_1^e + r_y c_1^e) && (5.5) \\ \Delta_2(\mathbf{r}) &= \frac{1}{2}(a_2^e + r_x b_2^e + r_y c_2^e) && (5.6) \\ \Delta_3(\mathbf{r}) &= \frac{1}{2}(a_3^e + r_x b_3^e + r_y c_3^e). && (5.7) \end{aligned}$$

Similarly, the equation for the area of the entire element can be derived by evaluating Equation 5.3 at point (x_1, y_1).

$$\Delta_e = \frac{1}{2}(b_1^e c_2^e - b_2^e c_1^e) \quad (5.8)$$

where

Finally given the equation for the area of the entire element, Δ_e the liner interpolation functions are[17]

$$\phi_1(\mathbf{r}) = \frac{\Delta_1(\mathbf{r})}{\Delta_e} \tag{5.9}$$

$$\phi_2(\mathbf{r}) = \frac{\Delta_2(\mathbf{r})}{\Delta_e} \tag{5.10}$$

$$\phi_3(\mathbf{r}) = \frac{\Delta_3(\mathbf{r})}{\Delta_e}. \tag{5.11}$$

These are the interpolation functions which would be used to in the formula given in Equation 5.1.

5.1.4 3D Elemental Interpolation

As mentioned previously, once the domain is discretized, one can approximate the unknown function within each element using the known values at the nodes and a linear combination of the interpolation functions. This derivation is similar to the $2D$ one with linear interpolations, but now the functions are in three dimensions. Thus, the unknown value u^e inside the element can be approximated by

$$u^e(x, y, z) = a^e + b^e x + c^e y + d^e z \tag{5.12}$$

The coefficients can be determined by enforcing Equation 5.12 at the four vertices. Denoting u_i^e as the value at node j on element e

[17]It should be noted that the linear interpolation functions derived here are equivalent to the Barycentric coordinates of a triangle.

$$u_1^e = a^e + b^e x_1^e + c^e y_1^e + d^e z_1^e \tag{5.13}$$

$$u_2^e = a^e + b^e x_2^e + c^e y_2^e + d^e z_2^e \tag{5.14}$$

$$u_3^e = a^e + b^e x_3^e + c^e y_3^e + d^e z_3^e \tag{5.15}$$

$$u_4^e = a^e + b^e x_4^e + c^e y_4^e + d^e z_4^e \tag{5.16}$$

from which the following is obtained:

$$a^e = \frac{1}{6V^e} \begin{vmatrix} u_1^e & u_2^e & u_3^e & u_4^e \\ x_1^e & x_2^e & x_3^e & x_4^e \\ y_1^e & y_2^e & y_3^e & y_4^e \\ z_1^e & z_2^e & z_3^e & z_4^e \end{vmatrix} = \frac{1}{6V^e}(a_1^e u_1^e + a_2^e u_2^e + a_3^e u_3^e + a_4^e u_4^e) \tag{5.17}$$

where

$$a_1^e = x_2(y_3 z_4 - y_4 z_3) + x_3(y_4 z_2 - y_2 z_4) + x_4(y_2 z_3 - y_3 z_2) \tag{5.18}$$

$$a_2^e = x_1(y_4 z_3 - y_3 z_4) + x_3(y_1 z_4 - y_4 z_1) + x_4(y_3 z_1 - y_1 z_3) \tag{5.19}$$

$$a_3^e = x_1(y_2 z_4 - y_4 z_2) + x_2(y_4 z_1 - y_1 z_4) + x_4(y_1 z_2 - y_2 z_1) \tag{5.20}$$

$$a_4^e = x_1(y_3 z_2 - y_2 z_3) + x_2(y_1 z_3 - y_3 z_1) + x_3(y_2 z_1 - y_1 z_2); \tag{5.21}$$

$$\tag{5.22}$$

$$b^e = \frac{1}{6V^e} \begin{vmatrix} 1 & 1 & 1 & 1 \\ u_1^e & u_2^e & u_3^e & u_4^e \\ y_1^e & y_2^e & y_3^e & y_4^e \\ z_1^e & z_2^e & z_3^e & z_4^e \end{vmatrix} = \frac{1}{6V^e}(b_1^e u_1^e + b_2^e u_2^e + b_3^e u_3^e + b_4^e u_4^e) \tag{5.23}$$

where

$$b_1^e = y_2(z_4 - z_3) + y_3(z_2 - z_4) + y_4(z_3 - z_2) \tag{5.24}$$

$$b_2^e = y_1(z_3 - z_4) + y_3(z_4 - z_1) + y_4(z_1 - z_3) \tag{5.25}$$

$$b_3^e = y_1(z_4 - z_2) + y_2(z_1 - z_4) + y_4(z_2 - z_1) \tag{5.26}$$

$$b_4^e = y_1(z_2 - z_3) + y_2(z_3 - z_1) + y_3(z_1 - z_2); \tag{5.27}$$

$$\tag{5.28}$$

$$c^e = \frac{1}{6V^e} \begin{vmatrix} 1 & 1 & 1 & 1 \\ x_1^e & x_2^e & x_3^e & x_4^e \\ u_1^e & u_2^e & u_3^e & u_4^e \\ z_1^e & z_2^e & z_3^e & z_4^e \end{vmatrix} = \frac{1}{6V^e}(c_1^e u_1^e + c_2^e u_2^e + c_3^e u_3^e + c_4^e u_4^e) \tag{5.29}$$

where

$$c_1^e = x_2(z_3 - z_4) + x_3(z_4 - z_2) + x_4(z_2 - z_3) \tag{5.30}$$

$$c_2^e = x_1(z_4 - z_3) + x_3(z_1 - z_4) + x_4(z_3 - z_1) \tag{5.31}$$

$$c_3^e = x_1(z_2 - z_4) + x_2(z_4 - z_1) + x_4(z_1 - z_2) \tag{5.32}$$

$$c_4^e = x_1(z_3 - z_2) + x_2(z_1 - z_3) + x_3(z_2 - z_1); \tag{5.33}$$

$$\tag{5.34}$$

and

$$d^e = \frac{1}{6V^e} \begin{vmatrix} 1 & 1 & 1 & 1 \\ x_1^e & x_2^e & x_3^e & x_4^e \\ y_1^e & y_2^e & y_3^e & y_4^e \\ u_1^e & u_2^e & u_3^e & u_4^e \end{vmatrix} = \frac{1}{6V^e}(d_1^e u_1^e + d_2^e u_2^e + d_3^e u_3^e + d_4^e u_4^e) \tag{5.35}$$

where

$$
\begin{aligned}
d_1^e &= x_2(y_4 - y_3) + x_3(y_2 - y_4) + x_4(y_3 - y_2) && (5.36)\\
d_2^e &= x_1(y_3 - y_4) + x_3(y_4 - y_1) + x_4(y_1 - y_3) && (5.37)\\
d_3^e &= x_1(y_4 - y_2) + x_2(y_1 - y_4) + x_4(y_2 - y_1) && (5.38)\\
d_4^e &= x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2). && (5.39)\\
& && (5.40)
\end{aligned}
$$

The volume of the element can be expressed as

$$
V^e = \frac{1}{6}\begin{vmatrix} 1 & 1 & 1 & 1 \\ x_1^e & x_2^e & x_3^e & x_4^e \\ y_1^e & y_2^e & y_3^e & y_4^e \\ z_1^e & z_2^e & z_3^e & z_4^e \end{vmatrix} = \frac{1}{6}\begin{vmatrix} x_1\left[y_2(z_4 - z_3) + y_3(z_2 - z_4) + y_4(z_3 - z_2)\right] + \\ x_2\left[y_1(z_3 - z_4) + y_3(z_4 - z_1) + y_4(z_1 - z_3)\right] + \\ x_3\left[y_1(z_4 - z_2) + y_2(z_1 - z_4) + y_4(z_2 - z_1)\right] + \\ x_4\left[y_1(z_2 - z_3) + y_2(z_3 - z_1) + y_3(z_1 - z_2)\right] \end{vmatrix}. \quad (5.41)
$$

Substituting the expressions for a^e, b^e, c^e and d^e back into Equation 5.12 yields

$$
u^e(x, y, z) = \sum_{j=1}^{4} \phi_j^e(x, y, z) u_j^e \quad (5.42)
$$

where the interpolation functions ϕ_j^e are defined by

$$
\phi_j^e(x, y, z) = \frac{1}{6V^e}(a_j^e + b_j^e x + c_j^e y + d_j^e z). \quad (5.43)
$$

General Form For reference Equation 3.28 is stated again

$$
\nabla \cdot \beta \nabla U_d(\mathbf{r}) - \lambda U_d(\mathbf{r}) + S(\mathbf{r}) = 0 \quad (5.44)
$$

The weighted residual for element e at node i can be derived by multiplying Equation 5.44 by the interpolation function and integrating over the domain of the element.

$$R_i^e = \int_\Omega \phi_i^e(\mathbf{r}) \left[\nabla \cdot \beta \nabla U_d^e(\mathbf{r})\right] \, d\Omega - \int_\Omega \phi_i^e(\mathbf{r}) \lambda U_d^e(\mathbf{r}) \, d\Omega + \int_\Omega \phi_i^e(\mathbf{r}) S^e(\mathbf{r}) \, d\Omega \qquad i = 1, 2, \ldots, \mathrm{n}. \tag{5.45}$$

Green's first identity can be used to convert an integral of a function times the gradient of a function across a lower dimension to a integral of higher order across a higher dimension (volume to area, area to line). For area (a) to volume (V) it is stated as

$$\int_s \phi(\nabla \Psi) \cdot da = \int_V [\phi \nabla^2 \Phi + (\nabla \phi) \cdot (\nabla \Psi)] dV \tag{5.46}$$

Using Green's first identity to convert Equation 5.45 to its weak form[18] results in

$$\begin{aligned} R_i^e \;=\; & -\int_\Omega \nabla \phi_i^e(\mathbf{r}) \cdot \beta \nabla U_d^e(\mathbf{r}) \, d\Omega - \int_\Omega \phi_i^e(\mathbf{r}) \lambda U_d^e(\mathbf{r}) \, d\Omega \\ & + \int_\Omega \phi_i^e(\mathbf{r}) S^e(\mathbf{r}) \, d\Omega + \int_{\Gamma_s'} \phi_i^e(\mathbf{r}) \beta \left(\frac{\partial}{\partial n^+} U_d^e(\mathbf{r}) \right) \, d\Gamma' \qquad \mathrm{i} = 1, 2, \ldots, \mathrm{n} \end{aligned} \tag{5.47}$$

where Γ_s' refers to the boundary of element e and the ∇ operator has been changed to a $\frac{\partial}{\partial n^+}$ since the interest lies with the U_d on the surface in the direction of the outward normal. Note that the last term in Equation 5.47 accounts for the continuity of flux between elements. Physically, this term should cancel out between elements, thus it is dropped completely unless an element resides on the boundary of the domain. The removal of this term is a common technique in finite element derivations called the weak boundary condition.

[18]The "weak" in the weak formulation comes from the fact that the differentiation requirement on the variable being solved has been weakened, since the equation now only contains a first derivative [52].

5.1.5 Boundary Conditions

For clarity, the diffusion boundary condition is stated again

$$\mu_{tr} U_d(\mathbf{r}) - \frac{2}{3}\frac{\partial}{\partial n^-} U_d(\mathbf{r}) + \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi} = 0. \tag{5.48}$$

When the boundary of an element lies on the boundary of the domain the last term in Equation 5.47 is replaced with the boundary condition given in Equation 5.48. In this case Equation 5.47 becomes

$$\begin{aligned} R_i^e \;=\; & -\int_\Omega \nabla\phi_i^e(\mathbf{r}) \cdot \beta \nabla U_d^e(\mathbf{r})\, d\Omega - \int_\Omega \phi_i^e(\mathbf{r}) \lambda U_d^e(\mathbf{r})\, d\Omega \\ & + \int_\Omega \phi_i^e(\mathbf{r}) S^e(\mathbf{r})\, d\Omega + \int_{\Gamma_s'} \phi_i^e(\mathbf{r}) \beta \left(\frac{\partial}{\partial n^+} U_d^e(\mathbf{r}) \right) d\Gamma' \\ & - \frac{3\mu_{tr}}{2} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \beta \left(U_d^e(\mathbf{r}) + \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi} \right) d\Gamma \qquad \mathrm{i} = 1, 2, \ldots, \mathrm{n}. \end{aligned} \tag{5.49}$$

Where Γ_s' is the internal boundary and Γ_s is the external. Note the sign change on the external boundary due to the reverse direction of the normal.

5.2 Matrix Construction

Since $U_d^e(\mathbf{r})$ can be interpolated from the values at the nodes, $U_{d_i}^e$, as in Equation 5.1 we substitute this equation into Equation 5.47

$$\begin{aligned} R_i^e \;=\; & -\beta \left[\int_\Omega \nabla\phi_i^e(\mathbf{r}) \cdot \nabla \{\phi^e(\mathbf{r})\}^{\mathrm{T}}\, d\Omega \right] \{U_d^e\} - \lambda \left[\int_\Omega \phi_i^e(\mathbf{r}) \{\phi^e(\mathbf{r})\}^{\mathrm{T}}\, d\Omega \right] \{U_d^e\} \\ & + \beta \left[\int_{\Gamma_s'} \phi_i^e(\mathbf{r}) \left(\frac{\partial}{\partial n^+} \{\phi^e(\mathbf{r})\}^{\mathrm{T}} \right) d\Gamma' \right] \{U_d^e\} + \int_\Omega \phi_i^e(\mathbf{r}) S^e(\mathbf{r})\, d\Omega \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathrm{i} = 1, 2, \ldots, \mathrm{n}. \end{aligned} \tag{5.50}$$

This equation can be written in matrix form as

$$\{R^e\} = [K^e]\{U_d^e\} - \{b^e\} \tag{5.51}$$

where $\{R^e\} = [R_1^e,\ R_2^e,\ \ldots,\ R_n^e]$,

$$K_{ij}^e = \int_\Omega \left[\beta \nabla \phi_i^e(\mathbf{r}) \cdot \nabla \phi_j^e(\mathbf{r}) + \lambda \phi_i^e(\mathbf{r})\, \phi_j^e(\mathbf{r})\right]\ d\Omega - \beta \int_{\Gamma_s'} \phi_i^e(\mathbf{r})\, \left[\hat{\mathbf{n}}^+ \cdot \nabla \phi_j^e(\mathbf{r})\right]\ d\Gamma' \tag{5.52}$$

and

$$b_i^e = \int_\Omega \phi_i^e(\mathbf{r}) S^e(\mathbf{r})\ d\Omega. \tag{5.53}$$

However, if any nodes ij lie on the boundary of the domain they must account for the boundary condition (Equation 5.48) by adding the following terms into K_{ij}^e and b_i^e

$$K_{ij}^s = \beta \frac{3\mu_{tr}}{2} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \phi_j^e(\mathbf{r})\ d\Gamma \tag{5.54}$$

and

$$b_i^s = -\beta \frac{3\mu_{tr}}{2} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi}\ d\Gamma. \tag{5.55}$$

To generate the matrix for the entire FEM mesh, the matrices from all the elements are summed together. To map the local elemental matrices to the global one we apply an $N \times 3$ matrix, $[P^e]$, to $[K^e]$ in the form

$$[K] = \sum_{e=1}^{M} [P^e] \cdot [K^e] \cdot [P^e]^T \tag{5.56}$$

where M is the number of elements and the elements of $[P^e]$ are zero except for elements P^e_{i0}, P^e_{j1}, and P^e_{k2}, which are 1 where i, j, and k are the global node numbers of local nodes 0, 1 and 2. Similarly the global vector $\{b\}$ is built off the local element source vector as

$$\{b\} = \sum_{e=1}^{M} [P^e] \cdot \{b^e\}. \tag{5.57}$$

5.2.1 Elemental Equations

Since $U^e_d(\mathbf{r})$ can be interpolated from the values at the nodes, $U^e_{d_i}$, as in Equation 5.1 this equation is substituted into Equation 5.47

$$\begin{aligned} R^e_i \;=\; & -\beta \left[\int_\Omega \nabla \phi^e_i(\mathbf{r}) \cdot \nabla \{\phi^e(\mathbf{r})\}^{\mathrm{T}} \, d\Omega \right] \{U^e_d\} - \lambda \left[\int_\Omega \phi^e_i(\mathbf{r}) \{\phi^e(\mathbf{r})\}^{\mathrm{T}} \, d\Omega \right] \{U^e_d\} \\ & + \beta \left[\int_{\Gamma'_s} \phi^e_i(\mathbf{r}) \left(\frac{\partial}{\partial n^+} \{\phi^e(\mathbf{r})\}^{\mathrm{T}} \right) \, d\Gamma' \right] \{U^e_d\} + \int_\Omega \phi^e_i(\mathbf{r}) S^e(\mathbf{r}) \, d\Omega \\ & \qquad i = 1, 2, \ldots, n. \end{aligned} \tag{5.58}$$

This equation can be written in matrix form

$$\{R^e\} = [K^e]\{U^e_d\} - \{b^e\} \tag{5.59}$$

where $\{R^e\} = [R^e_1,\ R^e_2,\ \ldots,\ R^e_n]$,

$$K^e_{ij} = \int_\Omega \left[\beta \nabla \phi^e_i(\mathbf{r}) \cdot \nabla \phi^e_j(\mathbf{r}) + \lambda \phi^e_i(\mathbf{r})\, \phi^e_j(\mathbf{r}) \right] \, d\Omega - \beta \int_{\Gamma'_s} \phi^e_i(\mathbf{r}) \left[\hat{\mathbf{n}}^+ \cdot \nabla \phi^e_j(\mathbf{r}) \right] \, d\Gamma' \tag{5.60}$$

and

$$b_i^e = \int_\Omega \phi_i^e(\mathbf{r}) S^e(\mathbf{r})\ d\Omega. \tag{5.61}$$

However, if any nodes ij lie on the boundary of the domain they must account for the boundary condition (Equation 5.48) by adding the following terms into K_{ij}^e and b_i^e

$$K_{ij}^s = \beta \frac{3\mu_{tr}}{2} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \phi_j^e(\mathbf{r})\ d\Gamma \tag{5.62}$$

and

$$b_i^s = -\beta \frac{3\mu_{tr}}{2} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi}\ d\Gamma. \tag{5.63}$$

Since the expansion, and therefore the weighting function associated with a node, spans all elements directly connected to the node, the weighted residual R_i associated with node i is a summation over the elements directly connected to node i. Therefore, Equation 5.58 may be expanded using the local and global relations and then sum it over each element to find that

$$\{R\} = \sum_{e=1}^{M} \{\overline{R}^e\} = \sum_{e=1}^{M} ([\overline{K}^e]\{\overline{U_d}^e\} - \{\overline{b}^e\}) \tag{5.64}$$

where the overbar is used to denote the vector that has been expanded or augmented. The system of equations can then be obtained by setting Equation 5.64 to zero

$$\sum_{e=1}^{M} ([\overline{K}^e]\{\overline{U_d}^e\} - \{\overline{b}^e\}) = \{0\} \tag{5.65}$$

5.3 Refinement of General FE Form to a Particular Dimension

Although the general form of the diffusion equation is derived in Section 5.1 some decisions must be made about the interpolation functions in order to apply it to a particular dimension. The following sections assume linear interpolation elements derived previously and apply the system to both two and three dimensions.

5.3.1 Two-Dimensions

This section contains the derivations for the coefficients of the elements of K_{ij}^e and b_i^e for the two dimensional triangular elements defined in Section 5.1.1. In the case of 2D elements, matrix $[K]$ is a 3×3 matrix while vector $\{b^e\}$ is a three element column vector. For clarity Equation 5.52 is presented again with the weak boundary condition in effect:

$$K_{ij}^e = \int_\Omega \left[\beta \nabla \phi_i^e(\mathbf{r}) \cdot \nabla \phi_j^e(\mathbf{r}) + \lambda \phi_i^e(\mathbf{r})\, \phi_j^e(\mathbf{r})\right]\, d\Omega. \tag{5.66}$$

In this case the linear interpolation functions defined by Equations 5.9, 5.10 and 5.11 are used; thus, the gradients can be evaluated analytically as follows:

$$\nabla \phi_i^e(\mathbf{r}) = \frac{1}{2\Delta^e}[b_i^e, c_i^e]. \tag{5.67}$$

The first term in Equation 5.66 can be evaluated by analytically calculating the dot product of the divergence operators

$$\nabla \phi_i^e(\mathbf{r}) \cdot \nabla \phi_j^e(\mathbf{r}) = \frac{1}{(2\Delta^e)^2}(b_i^e b_j^e + c_i^e e_j^e), \tag{5.68}$$

and using the following formula[19]

$$\iint_{\Omega^e} \left(\phi_i^e(\mathbf{r})\right)^l \left(\phi_j^e(\mathbf{r})\right)^m \left(\phi_k^e(\mathbf{r})\right)^n \, dx \, dy = \frac{l!\, m!\, n!}{(l+m+n+n+2)!} 2\Delta^e, \tag{5.69}$$

one can integrate Equation 5.66 analytically

$$K_{ij}^e = \frac{\beta}{4\Delta^e}(b_i^e b_j^e + c_i^e c_j^e) + \frac{\lambda \Delta^e}{12}\kappa^e(1+\delta_{ij}) \tag{5.70}$$

where

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

and

$$\begin{aligned} b_i^e &= \int_\Omega \phi_i^e(\mathbf{r}) S^e(\mathbf{r}) \, d\Omega & (5.71)\\ &= \int_\Omega \phi_i^e(\mathbf{r}) \left[\nu U_{ri}^e(\mathbf{r}) - \frac{3}{2\pi}\nabla \cdot \beta \mathbf{Q}_1^e(\mathbf{r})\right] \, d\Omega & (5.72)\\ &= \nu \int_\Omega \phi_i^e(\mathbf{r}) U_{ri}^e(\mathbf{r}) \, d\Omega - \frac{3\beta}{2\pi}\int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left[\hat{\mathbf{n}}^+ \cdot \mathbf{Q}_1^e(\mathbf{r})\right] \, d\Gamma. & (5.73) \end{aligned}$$

Where S^e is the source term on the diffusion equation derived in Equation 3.31. Notice the integral over the anisotropic $\mathbf{Q}_1(\mathbf{r})$ term has been replaced as a line integral using the divergence theorem. To solve the integral over the U_{ri} term we assume that we can expand $U_{ri}^e(\mathbf{r})$ inside the element as a combination of the values at the nodes

$$U_{ri}^e(\mathbf{r}) = \sum_{j=1}^{3} \phi_j^e(\mathbf{r}) U_{ri_j}^e = \{\phi^e(\mathbf{r})\}^{\mathrm{T}} \{U_{ri}^e\} = \{U_{ri}^e\}^{\mathrm{T}} \{\phi^e(\mathbf{r})\} \tag{5.74}$$

[19]This formula is derived for both 2D and 3D linear interpolation functions in Appendix A

Substituting this in for U^e_{ri} in Equation 5.73 yields

$$\begin{aligned} b^e_i &= \nu \int_\Omega \phi^e_i(\mathbf{r}) \left[U^e_{ri_i} \phi^e_i(\mathbf{r}) + U^e_{ri_j} \phi^e_j(\mathbf{r}) + U^e_{ri_k} \phi^e_k(\mathbf{r}) \right] d\Omega \\ &\qquad - \frac{3\beta}{2\pi} \int_{\Gamma_s} \phi^e_i(\mathbf{r}) \left[\hat{\mathbf{n}}^+ \cdot \mathbf{Q}^e_1(\mathbf{r}) \right] d\Gamma \end{aligned} \tag{5.75}$$

$$= \nu \Delta^e \left(\frac{U_{ri_i}}{6} + \frac{U_{ri_j}}{12} + \frac{U_{ri_k}}{12} \right) - \frac{3\beta}{2\pi} \int_{\Gamma_s} \phi^e_i(\mathbf{r}) \left[\hat{\mathbf{n}}^+ \cdot \mathbf{Q}^e_1(\mathbf{r}) \right] d\Gamma. \tag{5.76}$$

The second integral can be approximated by assuming that $\mathbf{Q}_1(\mathbf{r})$ varies linearly between two nodes on the element, a reasonable assumption if the elements are small. Thus, the value of $\mathbf{Q}_1(\mathbf{r})$ can be defined as a function of ξ varying from 0 to 1, or from node i to the distant node j:

$$\mathbf{F}(\xi, i, j) = \mathbf{Q}_1(\mathbf{r}_i)(1 - \xi) + \mathbf{Q}_1(\mathbf{r}_j)\xi \tag{5.77}$$

Using these assumptions the integral can be solved analytically as follows:

$$\begin{aligned} &\int_{\Gamma_s} \phi^e_i(\mathbf{r}) \left[\hat{\mathbf{n}}^+ \cdot \mathbf{Q}_1(\mathbf{r}) \right] d\Gamma \approx \\ &\qquad \int_0^1 (1 - \xi) \Big(|\Gamma_{ij}| \hat{\mathbf{n}}^+_{ij} \cdot \mathbf{F}(\xi, i, j) + |\Gamma_{ik}| \hat{\mathbf{n}}^+_{ij} \cdot \mathbf{F}(\xi, i, k) \Big) d\xi \end{aligned} \tag{5.78}$$

$$= |\Gamma_{ij}| \hat{\mathbf{n}}^+_{ij} \cdot \left(\frac{\mathbf{Q}_1(\mathbf{r}_i)}{3} + \frac{\mathbf{Q}_1(\mathbf{r}_j)}{6} \right) + |\Gamma_{ik}| \hat{\mathbf{n}}^+_{ik} \cdot \left(\frac{\mathbf{Q}_1(\mathbf{r}_i)}{3} + \frac{\mathbf{Q}_1(\mathbf{r}_k)}{6} \right) \tag{5.79}$$

where $\mathbf{r_i}$ is the point at node i, $\hat{\mathbf{n}}^+_{ij}$ refers to the outward normal of the edge whose endpoints consist of node i and j and $|\Gamma_{ij}|$ is the length of that edge. To summarize, b^e_i can be approximated as:

$$\begin{aligned} b^e_i &= \nu \Delta^e \left(\frac{U_{ri_i}}{6} + \frac{U_{ri_j}}{12} + \frac{U_{ri_k}}{12} \right) \\ &\quad - \frac{3\beta}{2\pi} \left[|\Gamma_{ij}| \hat{\mathbf{n}}^+_{ij} \cdot \left(\frac{\mathbf{Q}_1(\mathbf{r}_i)}{3} + \frac{\mathbf{Q}_1(\mathbf{r}_j)}{6} \right) + |\Gamma_{ik}| \hat{\mathbf{n}}^-_{ik} \cdot \left(\frac{\mathbf{Q}_1(\mathbf{r}_i)}{3} + \frac{\mathbf{Q}_1(\mathbf{r}_k)}{6} \right) \right] \end{aligned} \tag{5.80}$$

Solving for $\mathbf{Q}_1(\mathbf{r})$ in 2D With a Point Light Source The general form of $\mathbf{Q_1}(\mathbf{r})$ in 2D is

$$\mathbf{Q_1}(\mathbf{r}) = \frac{\mu_t}{2\pi} \int_{2\pi} \left[\int_{2\pi} p(\hat{\mathbf{s}}, \hat{\mathbf{s}}')\hat{\mathbf{s}}\, d\omega \right] I_{ri}(\mathbf{r}, \hat{\mathbf{s}}') d\omega' \tag{5.81}$$

assuming that the light source is defined as a "point" i.e. $I_{ri}(\mathbf{r}, \hat{\mathbf{s}}')$ is really multiplied by the Dirac delta function $\delta(\hat{\mathbf{x}} - \hat{\mathbf{l}})$ where $\hat{\mathbf{l}}$ is a unit vector in the direction pointing from the light source to point $\mathbf{r}$. Thus Equation 5.81 becomes

$$\mathbf{Q_1}(\mathbf{r}) = \frac{\mu_t}{2\pi} I_{ri}(\mathbf{r}, \hat{\mathbf{l}}) \left[\int_{2\pi} p(\hat{\mathbf{s}}, \hat{\mathbf{l}})\hat{\mathbf{s}}\, d\omega \right] \tag{5.82}$$

If the phase function is symmetric around $\hat{\mathbf{l}}$ then by definition components which are perpendicular to $\hat{\mathbf{l}}$ will cancel out. Thus one can simplify the calculation of the phase function by assuming the light is oriented in the $\hat{\mathbf{x}}$ direction, only accumulating the $\hat{\mathbf{x}}$ component of the phase function then rotating the result back to the $\hat{\mathbf{l}}$ direction.

$$\mathbf{Q_1}(\mathbf{r}) = \frac{\mu_t}{2\pi} I_{ri}(\mathbf{r}, \hat{\mathbf{l}}) \left[\int_0^{\pi} p(\theta) \cos(\theta) d\theta \right] \hat{\mathbf{l}}. \tag{5.83}$$

The inner integral can be solved using numerical methods.

2D Boundary Condition Since the elements in this case are triangular, one considers the boundary condition on the edge ij of an element which borders the outer domain. In this case the elemental matrix $[K^e]$ and source vector $\{b^e\}$ must include the boundary terms associated with such an edge:

$$K_{ij}^s = \beta \frac{3\mu_{tr}}{2} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \phi_j^e(\mathbf{r})\, d\Gamma \tag{5.84}$$

if i and j lie on Γ_s and

$$b_i^s = -\beta \frac{3\mu_{tr}}{2} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{2\pi} \, d\Gamma. \tag{5.85}$$

if i lies on Γ_s.

Since $\phi_i^e(\mathbf{r})$ varies linearly from 1 at node i to 0 at neighboring nodes Equation 5.84 can be written as

$$K_{ij}^s = |\Gamma_s| \beta \frac{3\mu_{tr}}{2} \int_0^1 \xi(1-\xi) \, d\xi = |\Gamma_s| \beta \frac{3\mu_{tr}}{2} \left(-\frac{\xi^3}{3} + \frac{\xi^2}{2} \right) \Big|_{\xi=1} = \beta \mu_{tr} \frac{|\Gamma_s|}{4} \tag{5.86}$$

unless $i = j$ then,

$$K_{ij}^s = \beta |\Gamma_s| \frac{3\mu_{tr}}{2} \int_0^1 \xi^2 \, d\xi = \beta |\Gamma_s| \frac{3\mu_{tr}}{2} \left(\frac{\xi^3}{3} \right) \Big|_{\xi=1} = \beta \mu_{tr} \frac{|\Gamma_s|}{2} \tag{5.87}$$

where $|\Gamma_s|$ is the length of the segment and $\xi = \phi_i^e(\mathbf{r})$ is the variable of substitution. Note that if node i or j does not lie on the boundary, then the interpolation function will be zero and thus no contribution should be added into K_{ij}.

The derivation for b_i^s is nearly identical to that of Equation 5.79 except that the formulation is slightly simpler since only one interpolation function is used. Again making the assumption that $\mathbf{Q}_1(\mathbf{r})$ varies linearly from the boundary start and end points, $\mathbf{r}_i$ and $\mathbf{r}_j$,

$$\begin{aligned} b_i^s &= -\beta \frac{3\mu_{tr}}{2} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{2\pi} \, d\Gamma & (5.88) \\ &= -\beta \frac{3\mu_{tr}}{2\pi} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r}) \, d\Gamma & (5.89) \\ &= -\beta \frac{3\mu_{tr}}{2\pi} |\Gamma_s| \hat{\mathbf{n}}_s^- \cdot \left(\frac{\mathbf{Q}_1(\mathbf{r}_i)}{3} + \frac{\mathbf{Q}_1(\mathbf{r}_j)}{6} \right) & (5.90) \end{aligned}$$

Note that if node i does not lie on the boundary, then the interpolation function will be zero and thus no contribution should be added into b_i.

To summarize, if node i lies on the boundary of the domain Γ_s then the following terms are added to the elemental matrix and source vector:

$$K_{ij}^s = \beta\mu_{tr}\frac{|\Gamma_s|}{4}(1+\delta_{ij}), \tag{5.91}$$

$$b_i^s = -\beta\mu_{tr}\frac{3}{2\pi}|\Gamma_s|\hat{\mathbf{n}}_s^- \cdot \left(\frac{\mathbf{Q}_1(\mathbf{r}_i)}{3} + \frac{\mathbf{Q}_1(\mathbf{r}_j)}{6}\right). \tag{5.92}$$

5.3.2 Three-Dimensions

This section contains the derivation for the coefficients of the elements of K_{ij}^e and b_i^e for the three dimensional triangular elements defined in Section 5.1.2. In the 3D case matrix $[K]$ is a 4×4 matrix while vector $\{b^e\}$ is a four element column vector. For clarity the formulation is shown in a dimensionless quantity again:

$$K_{ij}^e = \int_\Omega \left[\beta\nabla\phi_i^e(\mathbf{r}) \cdot \nabla\phi_j^e(\mathbf{r}) + \lambda\phi_i^e(\mathbf{r})\,\phi_j^e(\mathbf{r})\right]\, d\Omega. \tag{5.93}$$

Using the linear interpolation functions derived in Equation 5.43 the gradients can be evaluated directly as:

$$\nabla\phi_i^e(\mathbf{r}) = \frac{1}{6V^e}[b_i^e, c_i^e, d_i^e]. \tag{5.94}$$

Their dot product is

$$\nabla\phi_i^e(\mathbf{r}) \cdot \nabla\phi_j^e(\mathbf{r}) = \frac{1}{(6V^e)^2}(b_i^e b_j^e + c_i^e c_j^e + d_i^e d_j^e). \tag{5.95}$$

Using the following formula[20] to evaluate the last term in in Equation 5.93

[20]Derived in Appendix A.

$$\iiint_{V^e} \left(\phi_1^e(\mathbf{r})\right)^k \left(\phi_2^e(\mathbf{r})\right)^l \left(\phi_3^e(\mathbf{r})\right)^m \left(\phi_4^e(\mathbf{r})\right)^n \; dx\; dy\; dz = \frac{k!\; l!\; m!\; n!}{(k+l+m+n+n+3)!} 6V^e, \tag{5.96}$$

and Equation 5.95 to evaluate the first term, Equation 5.93 can be integrated analytically as:

$$K_{ij}^e = \left(\frac{\beta}{36V^e}(b_i^e b_j^e + c_i^e c_j^e + d_i^e d_j^e) + \frac{V^e}{20}\lambda(1+\delta_{ij}) \right) \tag{5.97}$$

where

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

and

$$\begin{aligned} b_i^e &= \int_\Omega \phi_i^e(\mathbf{r}) S^e(\mathbf{r})\; d\Omega & (5.98)\\ &= \int_\Omega \phi_i^e(\mathbf{r}) \left[\nu U_{ri}^e(\mathbf{r}) - \frac{3\beta}{4\pi} \nabla \cdot \mathbf{Q}_1^e(\mathbf{r}) \right]\; d\Omega & (5.99)\\ &= \nu \int_\Omega \phi_i^e(\mathbf{r}) U_{ri}^e(\mathbf{r})\; d\Omega - \frac{3\beta}{4\pi} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left[\hat{\mathbf{n}}^+ \cdot \mathbf{Q}_1^e(\mathbf{r}) \right]\; d\Gamma. & (5.100) \end{aligned}$$

Notice that the integral over the anisotropic $\mathbf{Q}_1(\mathbf{r})$ term has been replaced as a surface integral using the divergence theorem [70]. To solve the integral over the U_{ri} term, assume that$U_{ri}^e(\mathbf{r})$ can be expanded inside the element as a combination of the values at the nodes

$$U_{ri}^e(\mathbf{r}) = \sum_{j=1}^{4} \phi_j^e(\mathbf{r}) U_{ri_j}^e = \{\phi^e(\mathbf{r})\}^{\mathrm{T}} \{U_{ri}^e\} = \{U_{ri}^e\}^{\mathrm{T}} \{\phi^e(\mathbf{r})\}. \tag{5.101}$$

Substituting this in for U_{ri}^e in Equation 5.100 yields

$$
\begin{aligned}
b_i^e &= \nu \int_\Omega \phi_i^e(\mathbf{r}) \left[U_{ri_i}^e \phi_i^e(\mathbf{r}) + U_{ri_j}^e \phi_j^e(\mathbf{r}) + U_{ri_k}^e \phi_k^e(\mathbf{r}) + U_{ri_l}^e \phi_l^e(\mathbf{r}) \right] d\Omega \\
&\qquad - \frac{3\beta}{4\pi} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left[\hat{\mathbf{n}}^+ \cdot \mathbf{Q}_1^e(\mathbf{r}) \right] \, d\Gamma
\end{aligned} \tag{5.102}
$$

$$
\begin{aligned}
&= \nu V^e \left(\frac{U_{ri_i}}{10} + \frac{U_{ri_j}}{20} + \frac{U_{ri_k}}{20} + \frac{U_{ri_l}}{20} \right) \\
&\qquad - \frac{3\beta}{4\pi} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left[\hat{\mathbf{n}}^+ \cdot \mathbf{Q}_1^e(\mathbf{r}) \right] \, d\Gamma
\end{aligned} \tag{5.103}
$$

The second integral can be approximated using second order Gaussian Quadrature for triangles:

$$
b_i^e = \nu V^e \left(\frac{U_{ri_i}}{10} + \frac{U_{ri_j}}{20} + \frac{U_{ri_k}}{20} + \frac{U_{ri_l}}{20} \right) - \frac{\beta}{4\pi} \sum_{f=1}^{4} \left(\Delta_{s_f} \sum_{j=1}^{3} \phi_i^e(\mathbf{m}_j) \left[\hat{\mathbf{n}}_f^+ \cdot \mathbf{Q}_1^e(\mathbf{m}_j) \right] \right) \tag{5.104}
$$

Where f stands for the face on the element, Δ_{s_f} is the area of that face, $\mathbf{m}_j$ is the j^{th} midpoint on face f and $\hat{\mathbf{n}}_f^+$ is the normal on face f.

3D Source Term With a Point Light Source The solution of $\mathbf{Q}_1(\mathbf{r})$ in three dimensions with a point light source is the same as that of two dimensions (Equation 5.83) except for the normalization factor:

$$
\mathbf{Q_1}(\mathbf{r}) = \frac{\mu_t}{4\pi} I_{ri}(\mathbf{r}, \hat{\mathbf{l}}) \left[\int_{4\pi} p(\hat{\mathbf{s}}, \hat{\mathbf{l}}) \hat{\mathbf{s}} \, d\omega \right]. \tag{5.105}
$$

If the phase function is symmetric we can align the integration along the $\hat{\mathbf{x}}$ axis then project the solution back onto the light vector

$$
\mathbf{Q_1}(\mathbf{r}) = \frac{\mu_t}{4\pi} I_{ri}(\mathbf{r}, \hat{\mathbf{l}}) \left[\int_{4\pi} p(\hat{\mathbf{s}}, \hat{\mathbf{x}}) \hat{\mathbf{s}} \cdot \hat{\mathbf{x}} \, d\omega \right] \hat{\mathbf{l}}. \tag{5.106}
$$

Notice that the inner integral is simply the average cosine of the phase function from Equation 3.7. Thus, if p is the regular HG function

$$\int_{4\pi} p(\hat{\mathbf{s}}, \hat{\mathbf{x}})\hat{\mathbf{s}} \cdot \hat{\mathbf{x}}\, d\omega = g \tag{5.107}$$

and if p is the modified HG phase function the same integral becomes

$$\int_{4\pi} p(\hat{\mathbf{s}}, \hat{\mathbf{x}})\hat{\mathbf{s}} \cdot \hat{\mathbf{x}}\, d\omega = (1-\beta)g. \tag{5.108}$$

Thus,

$$\mathbf{Q_1}(\mathbf{r}) = \left[\frac{\mu_t}{4\pi} I_{ri}(\mathbf{r}, \hat{\mathbf{l}})(1-\beta)g\right]\hat{\mathbf{l}}. \tag{5.109}$$

3D Boundary Conditions To account for the no inward diffuse intensity boundary condition on a segment ij, the elemental matrix $[K^e]$ and source vector $\{b^e\}$ must include the boundary terms

$$K_{ij}^s = \beta\frac{3\mu_{tr}}{2}\int_{\Gamma_s} \phi_i^e(\mathbf{r})\phi_j^e(\mathbf{r})\, d\Gamma \tag{5.110}$$

if both i and j lie on Γ_s and

$$b_i^s = -\beta\frac{3\mu_{tr}}{2}\int_{\Gamma_s} \phi_i^e(\mathbf{r})\frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi}\, d\Gamma. \tag{5.111}$$

Equation 5.110 can be integrated analytically, resulting in

$$K_{ij}^s = \beta\mu_{tr}\frac{\Delta^s}{8}(1+\delta_{ij}). \tag{5.112}$$

Equation 5.111 can be integrated analytically using second order Gaussian quadrature

$$b_i^s = -\beta \frac{\mu_{tr}}{4\pi} \Delta_s \sum_{j=1}^{3} \phi_i^e(\mathbf{m}_j) \hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{m}_j) \tag{5.113}$$

where Δ_s is the area of the triangle defined by Γ_s and m_j is the j^{th} midpoint on the triangle.

5.4 Discussion

The finite element derivation presented in this chapter is applicable to light diffusion problems with triangular (2D) or tetrahedral (3D) arbitrary grids. Note that the solution calculated by this formulation will be accurate so long as the grid is fine enough. However, due to the computational complexity of solving fine grids, discretion needs to be applied. For example, it is not necessary to finely grid in places where the solution is low, however in sections of the grid where the solution is likely to change quickly (closer to the source or between the boundaries of different material types) a finer grid is required. This leads to the topic of Chapter 6 which contains a technique and implementation for adaptively refining grids using a method that does not require recalculation of the entire mesh and results in elements that are nicely shaped. Furthermore, to know where one must grid more or less finely a reliable per-error error estimation technique must be developed. Chapter 7 contains a novel technique for estimating the error inside a finite element solution where the source term is present. Finally, a 3D finite element implementation is presented in Chapter 8 and simulates the real world technique of near infrared subsurface scattering in a human arm.

CHAPTER 6

HANGING NODE *H*-ADAPTATION

"I ran into Isosceles. He had a great idea for a new triangle!"

— *Woody Allen.*

One of the main sources of error is due to the mesh density used to represent the geometry. Reliable and robust software for finite element simulations requires adaptive mesh refinement methods in which the software determines the mesh density without direct input from the user. This concept has two parts: (1) an *a posteriori* error estimate to determine where the error is in the solution and (2) a method to refine the mesh in regions of high error without reducing mesh quality and leaving regions with low error undisturbed. Significant work has been done on the first part (see for example [35] and later in Chapter 7). This chapter addresses the second part. The approach described here is based on a concept called hanging nodes [69] which has been recently used to allow nested refinement of elements in non-conforming grids. This method of finite element adaption lies in a class called h-adaptation, which attempts to refine the solution by subdividing elements with high error[21]. This chapter contains algorithms to efficiently subdivide the mesh to any level of refinement, to

[21] Other methods for finite element adaptation include r-adaptation, where the vertices of the mesh are moved, p-adaptation where higher order interpolation functions are used, and hp-adaptation where a combination of h and p adaptation are used simultaneously.

calculate the source distribution, and to pre-calculate the scatter interactions for light source updates.

6.1 Hanging Nodes / h-Adaptation

The hanging node refinement approach subdivides an element by introducing new nodes along the midpoints of its edges. The resultant mesh maintains the mesh quality of the original mesh, but it creates a nonconforming mesh since a large element now borders two smaller ones on the same face. Thus, the standard FEM basis functions now violate the required continuity conditions at the interface. To correct for the discontinuity, the basis functions in a large element are modified through the introduction of "ghost nodes" which do not contribute to the global solution (see Figure 6.1b). Instead, the ghost nodes are used as markers to identify which of the parent's basis functions should be shared across the element $[K^e]$ matrix.

In an unsubdivided element integrating $[K^e]$ into the global matrix $[K]$ simply requires a mapping from the local node numbers of element e to the global node numbers in the mesh. Thus the ij element in $[K^e]$ is directly copied to its respective element in $[K]$. In the case of a subdivided element which has a hanging node, the situation is more complex. In this case the hanging node has no global node number, instead the basis which contributes to that node is one half of one parent node, and one half of the other. Thus, mapping element ij from $[K^e]$ may involve distributing some weight to one global node and the rest to another.

Formally this is accomplished by modifying the $[P^e]$ matrix (introduced in Equation 5.56) such that the basis weights are shared correctly across the boundaries. The matrix $[V^e]$ is constructed for an unrefined element such that V^e_{ij} is 1 only if the

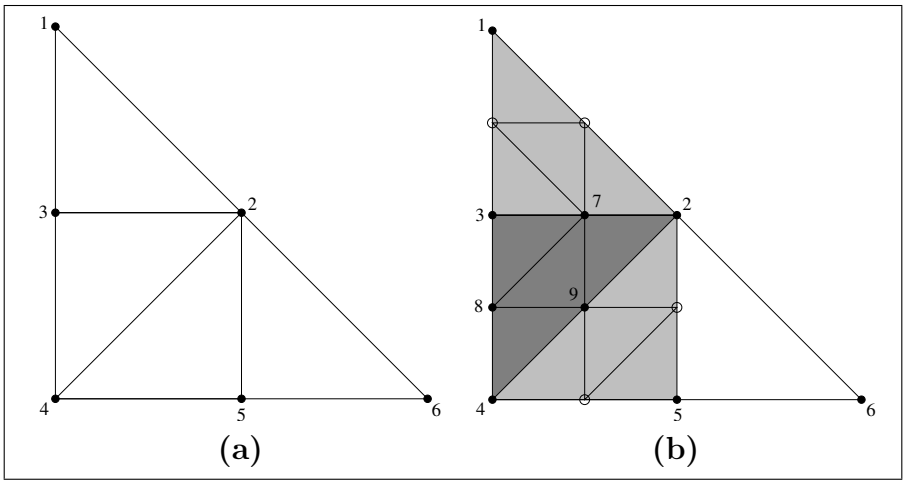


Figure 6.1: (a): A simple mesh with four elements. **(b)**: Element (2,3,4) is subdivided by introducing new nodes at the midpoints along the segments of the elements. To avoid a discontinuity, ghost nodes (indicated by unshaded circles) are inserted at the midpoints of the elements shaded in light gray.

local node j has the global index i. In the case of a subelement a node lies in the middle of the two parent nodes which generated it, thus its contribution should be half the contribution of the basis function from each of its parent nodes. To finally map the local $[K^e]$ elements to the global $[K]$ matrix, the matrix can be expanded as $[V^e] \cdot [K^e] \cdot [V^e]^T$ then added directly to $[K]$. The algorithm for creating the $[K]$ matrix in all cases is presented in the `Build_K` subroutine in Table 6.2, which is in turn based on the algorithm from [69].

One must be careful not to introduce new nodes when subdividing elements where the neighbor already contains a ghost node. For example, consider subdividing the element (2,5,6) in Figure 6.1(b). In this case there is already a ghost node defined on the edge (2,5) which is shared by the three subelements in the neighboring element. When element (2,5,6) is subdivided the three neighboring elements must be updated with the new non-ghost node. To efficiently search for previously created midpoints

a binary tree map is used where the elements are midpoints and the keys are the two parent nodes that create them. This way, when an element is subdivided it is simple to look up previously created midpoints in logarithmic time, then if no previous midpoint existed it is inserted the map.

6.2 Implementation

This section contains the details of an efficient implementation for the two most computationally expensive steps in calculating the solution of the diffusion equation in two dimensions, namely source calculation and matrix solution.

The source distribution is calculated from solving for $U_{ri}(\mathbf{r})$ at every node in the mesh (see Equation B.2) then integrating across the elements as shown in Equation 5.104. Calculating U_{ri} requires tracing a ray from the node to the light source and attenuating the source light based on the extinction cross section (σ_t) of the elements the ray passes through. Since it is possible to have a different material property in each element of the mesh an intersection must be calculated for each element the ray passes through.

To avoid expensive intersection tests with all the elements in the node, first all the elements which contain the source node are tested to determine which element the ray passes through. Then, one need only intersect the neighboring element across the face which the ray exits. One continues in this manner until a face is reached which lies on the boundary of the entire domain. This algorithm is detailed in Table 6.2 and illustrated in Figure 6.2. To efficiently determine neighbors and connected components, pointers to connected elements inside each node and neighbor pointers are stored inside each element's data structure. Table 6.3 demonstrates the large speedup

from using this algorithm as opposed to intersecting the ray with every element in the mesh.

Once the elemental global matrix and source vector are constructed, a linear system of the form $[K]\{U_d\} = \{b\}$ must be solved to calculate the diffuse light distribution. Fortunately the elemental matrix of the diffusion equation is positive-definite, thus Cholesky decomposition can be used to solve the system as was done with the finite difference matrix in Chapter 4.

6.3 Results and Discussion

The algorithm for mesh refinement, efficient 2D source calculation and Cholesky factorization is shown in Figure 6.3 and Figure 6.4 with timings and non-zero fill entries listed in Table 6.4. Figure 6.3 gives an overview of the base mesh and the initial source distribution and two sample plots (Figure 6.3c and Figure 6.3d) of the solved U_d given the coarse mesh in Figure 6.3 and the finest mesh in Figure 6.4a. As can be seen from Figure 6.3c, the solution on the course mesh has many discontinuities due to large elements along the boundary while the solution in Figure 6.3d is quite smooth due to extremely high refinement across the mesh.

Figure 6.4 demonstrates the refinement scheme given a full subdivision on Figure 6.4a, and several iterations of the same mesh selectively refined in Figure 6.4b. Finally, Figure 6.5 shows a plot of error versus number of elements in the mesh for both the fully subdivided mesh and the selectively refined mesh. The same level of error can be achieved using selective refinement with an order of magnitude fewer elements than the fully subdivided scheme.

```
Build_K([K]):
    for each parent element e do
        //N is the number of global nodes in the mesh
        [V^e]← is initialized to an empty sparse N × 3 matrix
        for all nodes i in e
            I ← the global node number of i
            V^e_Ii = 1
        call Assemble_Element([V^e],[K])
end //Build_K

Assemble_Element([V^e],[K]):
    if e has no children
        build the matrix [V^e] · [K^e] · [V^e]^T and add to [K]
    else
        for each subelement se in e do
            call Build_Vse([V^se])
            call Assemble_Element([V^se],[K])
end //Assemble_Element

Build_Vse([V^se]):
    //N is the number of global nodes in the mesh
    [V^se] ← is initialized to an empty sparse N × 3 matrix
    for each row I of [V^e] do
        for each vertex i of se do
            if i is a vertex of e then
                V^se_Ii = V^e_Ij
            else if i is a ghost node then
                (a, b) ← the edge on e which i is a midpoint
                V^se_Ii = 0.5 · (V^e_Ia + V^e_Ib)

    for each vertex i of se do
        if i is consistent and i is not a vertex of e then
            I ← the global node number of i
            V^se_Ii = 1
end //Build_Vse
```

Table 6.1: Algorithm to build $[K^e]$.

```
Calculate_Uri(node n, point l):
    r ← is the ray from node n to point l
    U_tmp ← is initialized to the intensity of l
    for each element e in n.connected_elements do
        if r intersects e at point p then
            attenuate U_tmp given the distance |n − p|
            e_n ← the neighbor of e on the edge of which p is located
            e ← e_n
            break out of loop
    while e is not NULL
        attenuate U_tmp given the distance |n − p|
        e_n ← the neighbor of the current e on the edge of which p is located
        e ← e_n
    U_ri ← U_tmp
```

Table 6.2: Linear time algorithm to calculate $U_{ri}(\mathbf{r})$

# of Elements	**Intersect Time (linear)**	**Intersect Time (enhanced)**
1.30×10^3	0.438s	0.015s
5.46×10^3	8.20s	0.078s
2.21×10^4	132.1s	0.312s
8.87×10^4	2090.0s	1.218s

Table 6.3: A comparison of timing results for calculation of U_{ri} given a linear time search of intersections and the optimized intersection test shown in Table 6.2.

6.4 Conclusions

This chapter contains a practical framework for solving light diffusion problems in two dimensions for the purpose of simulating light scatter in inhomogeneous materials. Specifically, an efficient implementation of hanging nodes to adaptively subdivide, while preserving the quality of the mesh, has been demonstrated; a method to economically compute the source distribution using element neighbor information has been shown; and finally it has been shown that a pre-factor of the finite element matrix can rapidly generate recalculations when the source distribution changes. Chapter 7 develops a per-element error estimation technique which along with the grid adaptation technique presented in this chapter can be used to adaptively refine the finite element mesh for better accuracy.

2.975in

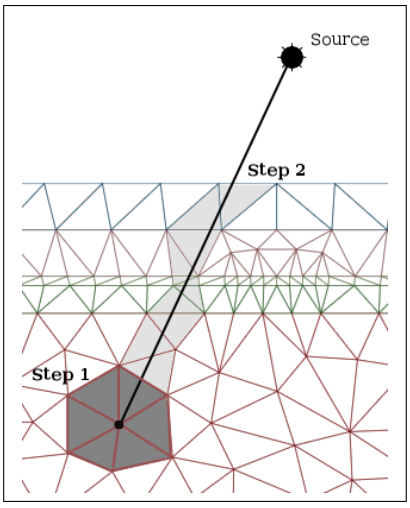


Figure 6.2: Calculating the source at a node (Figure 6.2). Step 1 (dark gray): test all the elements sharing the node and determine the edge of intersection. Step 2 (light gray): iteratively intersect neighbor elements until the boundary is reached.

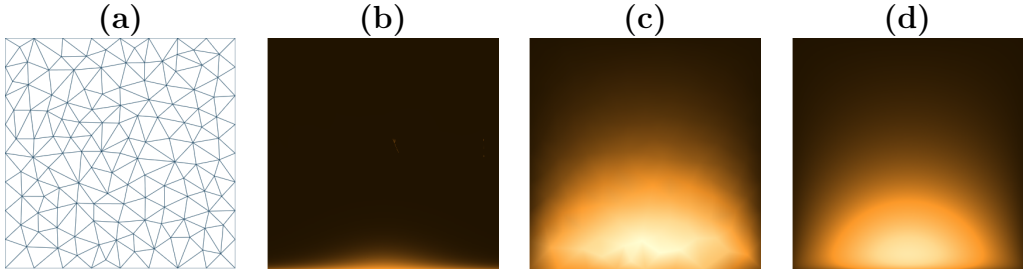


Figure 6.3: **(a)**: A square mesh containing 260 elements. **(b)**: The distribution of U_{ri} based on a point source beneath the mesh. **(c)**: The distribution of U_d given the source distribution from (b) on the mesh from (a). **(d)**: The distribution of U_d on the fully subdivided mesh shown in the last figure of Figure 6.4(a).

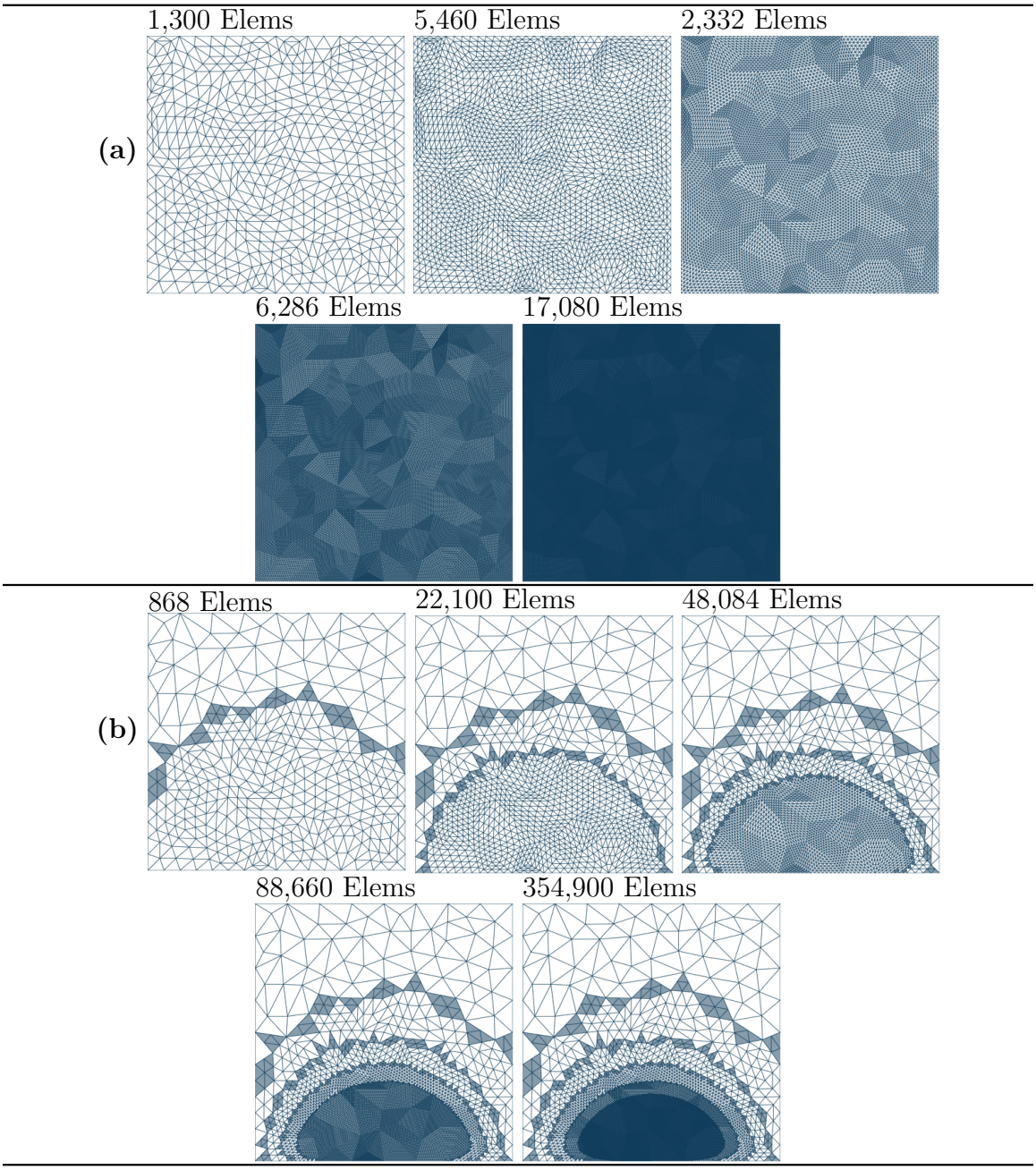


Figure 6.4: **(a)**: Five meshes which have been fully subdivided over successive iterations. **(b)**: The first mesh is adaptively subdivided on elements that contain nodes whose value of U_d is at least 10% of the maximum U_d in the solution given the source from Figure 6.3. The next column takes the previous mesh as input but subdivides elements which contain nodes of at least 20% of maximum. This continues up to to 50% of maximum for the last mesh. Darker elements indicate the presence of ghost nodes in those elements.

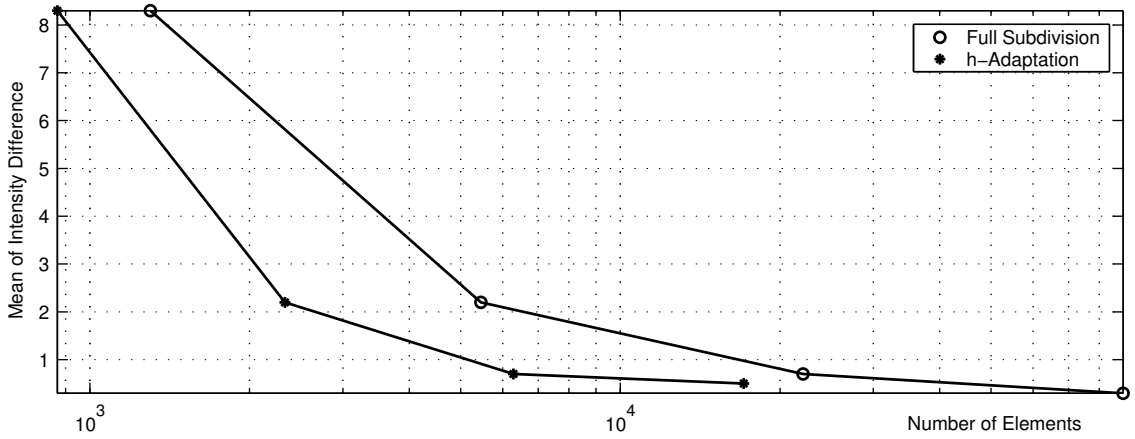


Figure 6.5: An error plot showing the average error intensity of U_d solved on both the fully and selectively refined meshes from Figure 6.4.

# of Elements	**Non-zeros in** $[K]$	**Non-zeros in** L	**Factor Time**	**Solution Time**
1.30×10^3	2.15×10^3	1.24×10^4	0.016s	<0.001s
5.46×10^3	8.45×10^3	2.59×10^6	0.032s	<0.001s
2.21×10^4	3.35×10^4	3.28×10^5	0.171s	0.016s
8.87×10^4	1.34×10^5	1.59×10^6	0.859s	0.047s
3.55×10^5	5.35×10^5	7.29×10^6	4.359s	0.218s

Table 6.4: Progression for timings and non-zero fill rate as mesh size is increased. These numbers were generated from the five iterations on the full subdivision mesh in Figure 6.4(a). The non-zeros in $[K]$ increase roughly linearly with the number of elements in the mesh. The non-zeros in the L factor increase roughly by an $O(n \log n)$ factor, while the factor and solution time increase approximately as $O(n^2)$.

CHAPTER 7

ERROR ESTIMATION

"To err is human, and to blame it on a computer is even more so."

— *Robert Orben.*

Not surprisingly there are numerous potential sources for error in any numerical simulation which can range from errors in the mathematical model, human error in the implementation, to inherent instabilities in the numerical solution itself. This chapter is about an important aspect of the error in the numerical realizations of the diffusion approximation using the FEM. Specifically, the error in the solution as expressed in Equation 5.56 is caused by discretizing the domain. A coarse discretization causes two sources of error: (1) the error associated with linearly interpolating the source term even though it propagates exponentially across the element; and (2) the "jump discontinuity" which occurs due to the derivative of the interpolation functions on the shared boundary of elements. The latter is a manifestation of the use of discrete elements. Both these errors can be reduced by reducing the size of the elements or increasing the order of the interpolation functions across such elements.

Although one can calculate the residual of a linear system by substituting the solution back into the original equation, this does not give any information on the local region where the error may be high. A more useful class of techniques attempts

to estimate the error at a local scale, specifically in an element. These techniques generally require the user to first generate a global solution, then use the global solution on a per element basis to attempt to estimate the error.

A currently popular error estimation technique, the self-equilibrated residual and complementary *a posteriori* error estimator, fails for this type of problem; specifically where the source is present in an element. This chapter first describes the derivation for this *a posteriori* error estimation technique in Section 7.1 for the diffusion equation and discusses why it fails to accurately estimate the error. In Section 7.2 a novel alternative error estimator is proposed based on the local solution using a Green's function kernel. This alternative is stable and unlike current *a posteriori* estimators, it accounts for the source distribution across the local element.

7.1 Self-Equilibrated Residual and Complementary *A Posteriori* Error Estimator

This section contains the derivation for a per-element self-equilibrated residual and complementary *a posteriori* error analysis technique. In combination with h-adaptation hanging node subdivision, presented in Chapter 6, this technique can be used to adaptively subdivide a grid in areas that have error above a tolerance level without subdividing neighboring elements which have error below the set tolerance.

7.1.1 Derivation

The derivation starts with the weak form of the non-boundary condition weighted residual of the diffusion equation derived using Galerkin's method (from Equation 5.47). It is again listed for clarity:

$$\begin{aligned} R_i^e &= -\int_\Omega \nabla\phi_i^e(\mathbf{r})\cdot\nabla U_d(\mathbf{r})\,d\Omega - \int_\Omega \phi_i^e(\mathbf{r})\kappa U_d(\mathbf{r})\,d\Omega \\ &\quad + \int_\Omega \phi_i^e(\mathbf{r})S(\mathbf{r})\,d\Omega + \int_{\Gamma_s} \phi_i^e(\mathbf{r})\left(\frac{\partial}{\partial\, n^+}U_d(\mathbf{r})\right)\,d\Gamma. \end{aligned} \tag{7.1}$$

However, for cases where the elements are internal, the integral along the boundary will cancel out and the resulting equation becomes:

$$R_i^e = -\int_\Omega \nabla\phi_i^e(\mathbf{r})\cdot\nabla U_d(\mathbf{r})\,d\Omega - \int_\Omega \phi_i^e(\mathbf{r})\kappa U_d(\mathbf{r})\,d\Omega + \int_\Omega \phi_i^e(\mathbf{r})S(\mathbf{r})\,d\Omega. \tag{7.2}$$

Let $\hat{U}_d$ be the solution from the finite element method when applied to Equation 7.2, then $\hat{U}_d$ satisfies:

$$R_i^e = -\int_\Omega \nabla\phi_i^e(\mathbf{r})\cdot\nabla \hat{U}_d(\mathbf{r})\,d\Omega - \int_\Omega \phi_i^e(\mathbf{r})\kappa \hat{U}_d(\mathbf{r})\,d\Omega + \int_\Omega \phi_i^e(\mathbf{r})S(\mathbf{r})\,d\Omega = 0. \tag{7.3}$$

Working backwards from the original finite element derivation presented in Chapter 5, Green's first identity is applied to Equation 7.3

$$\begin{aligned} R_i^e &= \int_\Omega \phi_i^e(\mathbf{r})\nabla\cdot\nabla\hat{U}_d(\mathbf{r})\,d\Omega - \int_{\Gamma_s} \phi_i^e(\mathbf{r})\left(\frac{\partial}{\partial n^+}\hat{U}_d(\mathbf{r})\right)\,d\Gamma \\ &\quad - \int_\Omega \phi_i^e(\mathbf{r})\kappa\hat{U}_d(\mathbf{r})\,d\Omega + \int_\Omega \phi_i^e(\mathbf{r})S(\mathbf{r})\,d\Omega = 0. \end{aligned} \tag{7.4}$$

Defining $r(\mathbf{r}) = \nabla\cdot\nabla\hat{U}_d(\mathbf{r}) - \kappa\hat{U}_d(\mathbf{r}) + S(\mathbf{r})$ as the residual which is not necessarily zero, simplifies Equation 7.4 to

$$R_i^e = \int_\Omega \phi_i^e(\mathbf{r})r(\mathbf{r})\,d\Omega - \int_{\Gamma_s} \phi_i^e(\mathbf{r})\left(\frac{\partial}{\partial n^+}\hat{U}_d(\mathbf{r})\right)\,d\Gamma = 0. \tag{7.5}$$

Setting Equation 7.5 equal to Equation 7.3 yields

$$\begin{aligned} R_i^e &= -\int_\Omega \nabla\phi_i^e(\mathbf{r}) \cdot \nabla\hat{U}_d(\mathbf{r})\ d\Omega - \int_\Omega \phi_i^e(\mathbf{r})\kappa\hat{U}_d(\mathbf{r})\ d\Omega + \int_\Omega \phi_i^e(\mathbf{r})S(\mathbf{r})\ d\Omega \\ &= \int_\Omega \phi_i^e(\mathbf{r})r(\mathbf{r})\ d\Omega - \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left(\frac{\partial}{\partial n^+}\hat{U}_d(\mathbf{r})\right)\ d\Gamma. \end{aligned} \tag{7.6}$$

To get the general form of the **non-boundary** case, subtract Equation 7.6 from Equation 7.2

$$-\int \nabla\phi_i^e(\mathbf{r})\cdot\nabla e(\mathbf{r})d\Omega - \int \phi_i^e(\mathbf{r})\kappa e(\mathbf{r})\ d\Omega = \int \phi_i^e(\mathbf{r})r(\mathbf{r})\ d\Omega - \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left(\kappa\frac{\partial}{\partial n^+}\hat{U}_d(\mathbf{r})\right)\ d\Gamma \tag{7.7}$$

where $e(\mathbf{r}) = \hat{U}_d(\mathbf{r}) - U_d(\mathbf{r})$.

The **boundary** case is derived from subtracting Equation 7.6 from Equation 7.1 and replacing the derivative with the boundary condition given in Equation 3.36

$$\begin{aligned} \int \phi_i^e(\mathbf{r})r(\mathbf{r})\ d\Omega - \int \nabla\phi_i^e(\mathbf{r}) \cdot \nabla e(\mathbf{r})d\Omega - \int \phi_i^e(\mathbf{r})\kappa e(\mathbf{r})\ d\Omega \\ - \frac{3\mu_{tr}}{2}\int_{\Gamma_s} \phi_i^e(\mathbf{r})e(\mathbf{r})\ d\Gamma = \int \phi_i^e(\mathbf{r})r(\mathbf{r})\ d\Omega \end{aligned} \tag{7.8}$$

where the source term S in the residual r now also includes the boundary source term containing $\mathbf{Q}_1$ from Equation 3.36.

7.1.2 Application to Two-Dimensional Diffusion Equation

For the non-boundary case, Equation 7.7 is treated as a per-element error equation even though there is some coupling between elements, namely the boundary integral in Equation 7.1. Note that Equation 7.7 is a finite element equation in itself where the unknown quantity in this case is e.

The residual term r can be calculated from the known values $\hat{U}_d$ but can be simplified by noting that since the original interpolation function ϕ is linear the double gradient $\nabla \cdot \nabla\hat{U}_d$ is zero.

The weighted residual from Equation 7.7 for any node i of the element is given by

$$\begin{aligned} R_i^e &= \int \nabla\phi_i^e(\mathbf{r}) \cdot \nabla e(\mathbf{r})\, d\Omega + \int \phi_i^e(\mathbf{r})\kappa e(\mathbf{r})\, d\Omega + \int \phi_i^e(\mathbf{r}) r(\mathbf{r})\, d\Omega \\ &\quad - \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left(\frac{\partial}{\partial n^+} \hat{U}_d(\mathbf{r}) \right) d\Gamma. \end{aligned} \tag{7.9}$$

This system of equations can be formulated to solve for e_j^e, the error at the nodes. This system of equations has the same $[K]$ matrix as Equation 5.70 however, the source term b_i^e in Equation 5.80 changes to

$$b_i^e = -\int \phi_i^e(\mathbf{r}) r(\mathbf{r})\, d\Omega + \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left(\frac{\partial}{\partial n^+} \hat{U}_d(\mathbf{r}) \right) d\Gamma \tag{7.10}$$

$$= -S_i^e + \int \phi_i^e(\mathbf{r})\kappa \hat{U}_d(\mathbf{r})\, d\Omega + \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left(\frac{\partial}{\partial n^+} \hat{U}_d(\mathbf{r}) \right) d\Gamma \tag{7.11}$$

given that $r(\mathbf{r}) = \nabla \cdot \nabla \hat{U}_d(\mathbf{r}) - \kappa \hat{U}_d(\mathbf{r}) + S(\mathbf{r})$ and (originally from Equation 5.76):

$$\begin{aligned} S_i^e &= \nu \Delta^e \left(\frac{U_{ri_i}}{6} + \frac{U_{ri_j}}{12} + \frac{U_{ri_k}}{12} \right) \\ &\quad - \frac{3\beta}{2\pi} \left[|\Gamma_{ij}| \hat{\mathbf{n}}_{ij}^+ \cdot \left(\frac{\mathbf{Q}_1(\mathbf{r}_i)}{3} + \frac{\mathbf{Q}_1(\mathbf{r}_j)}{6} \right) + |\Gamma_{ik}| \hat{\mathbf{n}}_{ik}^+ \cdot \left(\frac{\mathbf{Q}_1(\mathbf{r}_i)}{3} + \frac{\mathbf{Q}_1(\mathbf{r}_k)}{6} \right) \right]. \end{aligned} \tag{7.12}$$

The term $\int \phi_i^e(\mathbf{r})\kappa \hat{U}_d(\mathbf{r})\, d\Omega$ in Equation 7.11 can be expanded in the same manner as was done for Equation 5.103 by assuming that $\hat{U}_d(\mathbf{r})$ can be interpolated inside the element by a linear combination of the values at the nodes. Applying this results in:

$$b_i^e = -S_i^e + \kappa \Delta^e \left(\frac{\hat{U}_{d_i}^e}{6} + \frac{\hat{U}_{d_j}^e}{12} + \frac{\hat{U}_{d_k}^e}{12} \right) + \int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left(\frac{\partial}{\partial n^+} \hat{U}_d(\mathbf{r}) \right) d\Gamma \tag{7.13}$$

where $\hat{U}_{d_i}^e$ is the value of $\hat{U}_d$ at node i in element e and S_i^e is the value of S at node i in element e. The last term in Equation 7.11 can be partially calculated analytically. First the partial derivative is expanded in two dimensions

$$\frac{\partial}{\partial n^+} = \mathbf{n}^+ \cdot \mathbf{x}\frac{\partial}{\partial x} + \mathbf{n}^+ \cdot \mathbf{y}\frac{\partial}{\partial y}. \tag{7.14}$$

Then given that the term $\hat{U}_d$ can be written as a linear combination of nodal values:

$$\hat{U}_d(\mathbf{r}) = \sum_{j=1}^{3} \hat{U}^e_{d_j}\phi^e_j(\mathbf{r}), \tag{7.15}$$

Equation 7.14 can be rewritten as,

$$\frac{\partial}{\partial n^+}\hat{U}_d(\mathbf{r}) = \sum_{j=1}^{3} \hat{U}^e_{d_j}\left(\mathbf{n}^+ \cdot \mathbf{x}\frac{\partial \phi^e_j(\mathbf{r})}{\partial x} + \mathbf{n}^+ \cdot \mathbf{y}\frac{\partial \phi^e_j(\mathbf{r})}{\partial y}\right). \tag{7.16}$$

Substituting Equation 7.16 into the last term in Equation 7.13 yields

$$\int_{\Gamma_s} \phi^e_i(\mathbf{r})\left(\frac{\partial}{\partial n^+}\hat{U}_d(\mathbf{r})\right)d\Gamma = \int_{\Gamma_s} \phi^e_i(\mathbf{r})\left(\sum_{j=1}^{3} \hat{U}^e_{d_j}\left(\mathbf{n}^+ \cdot \mathbf{x}\frac{\partial \phi^e_j(\mathbf{r})}{\partial x} + \mathbf{n}^+ \cdot \mathbf{y}\frac{\partial \phi^e_j(\mathbf{r})}{\partial y}\right)\right)d\Gamma. \tag{7.17}$$

Since the interpolation functions are linear, they can be explicitly differentiated

$$\frac{\partial \phi^e_1}{\partial x} = \frac{b^e_1}{2\Delta_e} \tag{7.18}$$

$$\frac{\partial \phi^e_2}{\partial x} = \frac{b^e_2}{2\Delta_e} \tag{7.19}$$

$$\frac{\partial \phi^e_3}{\partial x} = \frac{b^e_3}{2\Delta_e} \tag{7.20}$$

$$\frac{\partial \phi^e_1}{\partial y} = \frac{c^e_1}{2\Delta_e} \tag{7.21}$$

$$\frac{\partial \phi^e_2}{\partial y} = \frac{c^e_2}{2\Delta_e} \tag{7.22}$$

$$\frac{\partial \phi^e_3}{\partial y} = \frac{c^e_3}{2\Delta_e} \tag{7.23}$$

If node i does not lie on the boundary Γ_s then the contribution from that integral will be zero, otherwise the integral in Equation 7.17 can be modified as:

$$\int_{\Gamma_s} \phi_i^e(\mathbf{r}) \left(\frac{\partial}{\partial n} \hat{U}_d(\mathbf{r}) \right) d\Gamma \tag{7.24}$$

$$= |\Gamma_s| \int_0^1 \xi \left(\sum_{j=1}^{3} \hat{U}_{d_j} \left(\mathbf{n}_s^+ \cdot \mathbf{x} \frac{\partial \phi_j^e(\xi)}{\partial x} + \mathbf{n}_s^+ \cdot \mathbf{y} \frac{\partial \phi_j^e(\xi)}{\partial y} \right) \right) d\xi \tag{7.25}$$

$$= |\Gamma_s| \int_0^1 \xi \left(\sum_{j=1}^{3} \hat{U}_{d_j} \left(\mathbf{n}_s^+ \cdot \mathbf{x} \frac{b_j^e}{2\Delta_e} + \mathbf{n}_s^+ \cdot \mathbf{y} \frac{c_j^e}{2\Delta_e} \right) \right) d\xi \tag{7.26}$$

$$= \frac{1}{2\Delta_e} |\Gamma_s| \left(\sum_{j=1}^{3} \hat{U}_{d_j} \left((\mathbf{n}_s^+ \cdot \mathbf{x}) b_j^e + (\mathbf{n}_s^+ \cdot \mathbf{y}) c_j^e \right) \right) \int_0^1 \xi \, d\xi \tag{7.27}$$

$$B(e, \Gamma_s) = \frac{1}{4\Delta_e} |\Gamma_s| \left(\sum_{j=1}^{3} \hat{U}_{d_j} \left((\mathbf{n}_s^+ \cdot \mathbf{x}) b_j^e + (\mathbf{n}_s^+ \cdot \mathbf{y}) c_j^e \right) \right) \tag{7.28}$$

However, the error along the boundary is not just contributed from the current element e but also the element neighboring e along boundary Γ_s, e_s. Thus, the total error along the boundary is actually $B(e, \Gamma_s) + B(e_s, \Gamma_s)$. Since this error arises from the boundary, and not the element, the error is shared between the two elements based on the relative sizes of the elements. Thus the total amount of error on the boundary that is attributed to element e is

$$\left[B(e, \Gamma_s) + B(e_s, \Gamma_s) \right] \frac{\Delta^e}{\Delta^e + \Delta^{e_s}} \tag{7.29}$$

To summarize, the $[K]$ matrix for the error estimator is identical to the original finite element matrix:

$$K_{ij}^e = \left(\frac{1}{4\Delta^e} (b_i^e b_j^e + c_i^e c_j^e) + \frac{\Delta^e}{12} \kappa^e (1 + \delta_{ij}) \right). \tag{7.30}$$

where

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

and the b vector is

$$b_i^e = -S_i^e + \kappa\Delta^e \left(\frac{\hat{U}_{d_i}^e}{6} + \frac{\hat{U}_{d_j}^e}{12} + \frac{\hat{U}_{d_k}^e}{12} \right) + \left[\left[B(e, \Gamma_s) + B(e_s, \Gamma_s) \right] \frac{\Delta^e}{\Delta^e + \Delta^{e_s}} \forall \left(\Gamma_s \text{which touch node } i \right) \right] \tag{7.31}$$

Elemental Boundary Conditions

In the case of a domain boundary element the weighted residual will be

$$R_i = \int \nabla\phi_i^e(\mathbf{r}) \cdot \nabla e(\mathbf{r}) d\Omega + \int \phi_i^e(\mathbf{r}) \kappa e(\mathbf{r}) \, d\Omega + \frac{3\mu_{tr}}{2} \int_{\Gamma_s} \phi_i^e(\mathbf{r}) e(\mathbf{r}) \, d\Gamma + \int \phi_i^e(\mathbf{r}) r(\mathbf{r}) \, d\Omega \tag{7.32}$$

where the term $r(\mathbf{r})$ now contains a boundary term as well. The modifications to $[K]$ and $\{b\}$ are

$$K_{ij}^s = \mu_{tr} \frac{|\Gamma_s|}{4} (1 + \delta_{ij}), \tag{7.33}$$

$$b_i^s = -\mu_{tr} \frac{3}{2\pi} |\Gamma_s| \hat{\mathbf{n}}_s^- \cdot \left(\frac{\mathbf{Q}_1(\mathbf{r}_i)}{3} + \frac{\mathbf{Q}_1(\mathbf{r}_j)}{6} \right). \tag{7.34}$$

7.1.3 Instability in the Presence of a Source

Although the self-equilibrated residual and complementary *a posteriori* error estimator technique has been used to predict the error in stress and strain calculations [51] as well as many other domains, it seems to be unsuitable for the diffusion equation as presented in this dissertation. The major difference between the domains is the distribution of the source. In many domains the source is distributed across the outer

boundary of the domain. However, in the light diffusion problem the source is the reduced incident intensity term (U_{ri}) which is prevalent throughout the volume.

In the elements where the source term is large the error prediction technique described previously does not converge as elements are subdivided. The author has experimentally verified this by artificially "cutting off" the source term in certain elements and observed that the self-equilibrated residual and complementary *a posteriori* error estimator technique accurately predicts the error in those elements once the source has been removed (see Figure 7.1 for such an element and the results of the error estimator in Figure 7.2). Unfortunately although this verifies the instability of the method presented here it does not offer a solution as removing the source term from any element changes the overall solution. A solution to this problem is proposed in the next section.

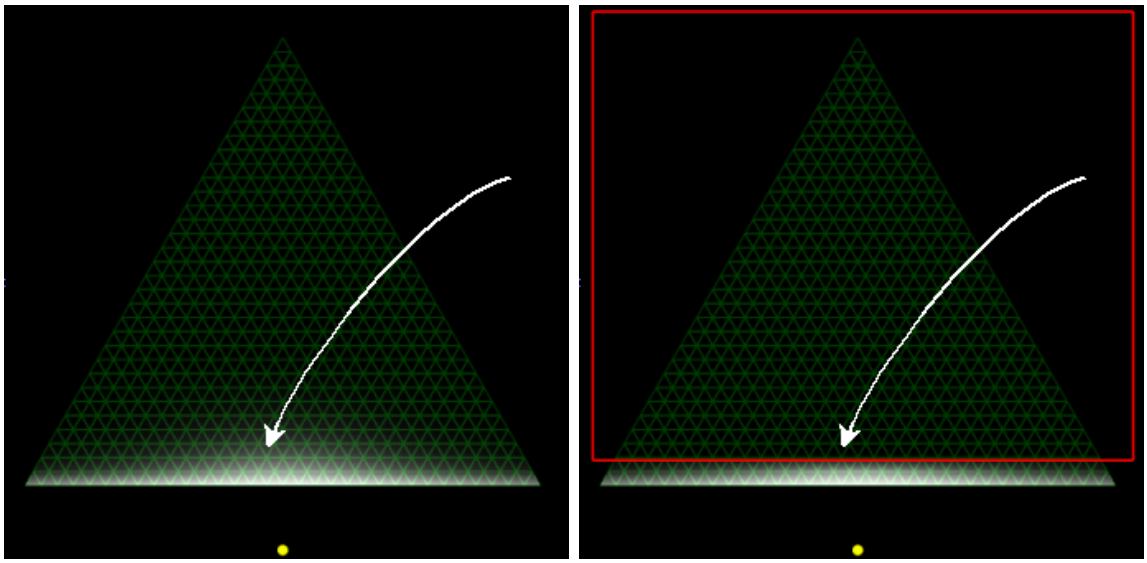


Figure 7.1: Arrow indicates element used for error estimation throughout this section. The left figure shows the original distribution of U_{ri} while the right shows the region which artificially cuts off U_{ri} to test the self-equilibrated residual and complementary *a posteriori* error estimator.

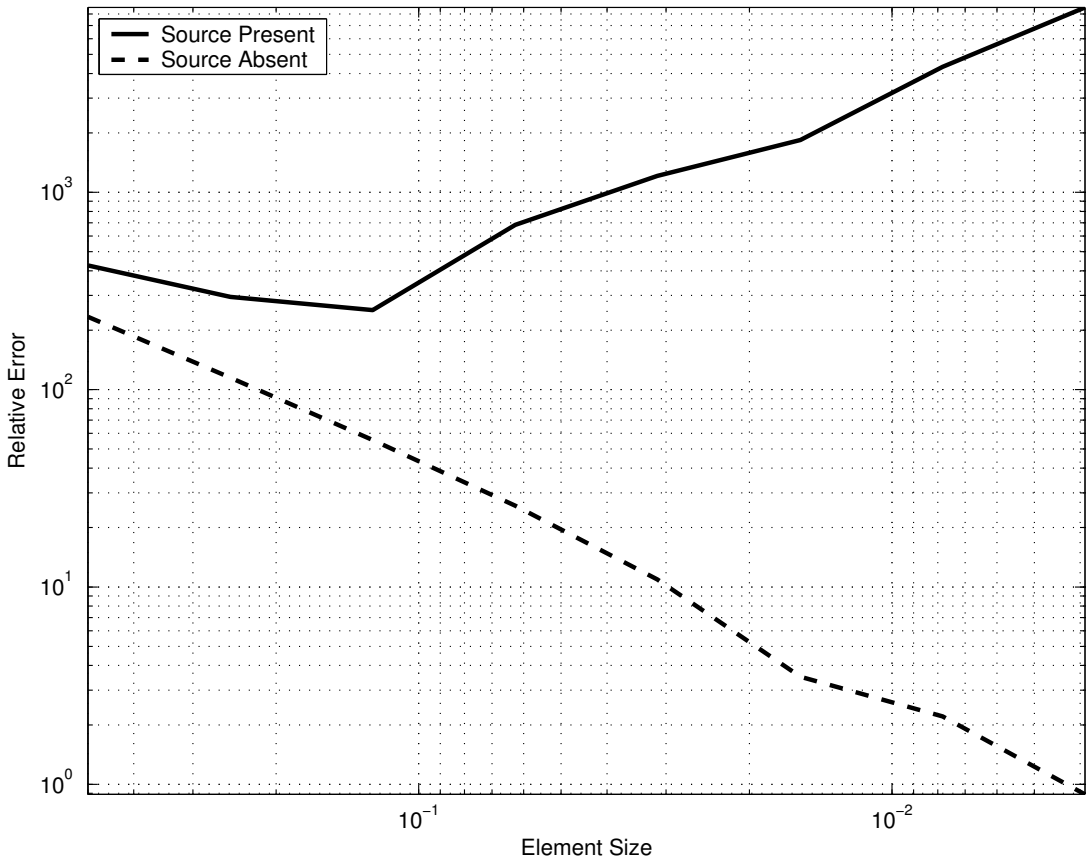


Figure 7.2: This plot demonstrates the instability of the *a posteriori* error estimator described in Section 7.1.2. In this experiment the normalized error (the calculated error divided by the numerical value of U_{ri}) is calculated for the element in Figure 7.1 calculated per element in both the presence of a source (solid line) and its artificially removed absence (dashed line). The error grows without bound in the presence of a source, but is otherwise well behaved without it. The "element size" axis is the length in cm of the edge of the element (an equilateral triangle).

7.2 Error Estimation: Method of Moments and Green's Function

In this section an error estimator is developed using an integral equation which solves the differential equation through the integration of a Green's function kernel.

As an example, consider the following differential equation

$$L(\mathbf{r})u(\mathbf{r}) = f(\mathbf{r}) \tag{7.35}$$

where $L(\mathbf{r})$ is the differential operator, $u(\mathbf{r})$ is an unknown function and $f(\mathbf{r})$ is a known source function. A solution to Equation 7.35 can be written as

$$u(\mathbf{r}) = L^{-1}(\mathbf{r})f(\mathbf{r}) \tag{7.36}$$

where $L^{-1}(\mathbf{r})$ is the inverse of $L(\mathbf{r})$. Thus, the inverse operator in Equation 7.36 can be defined using a Green's function as

$$L^{-1}(\mathbf{r})f(\mathbf{r}) = \int_{\Omega} G(\mathbf{r};\mathbf{r}')f(\mathbf{r}')d\mathbf{r}' \tag{7.37}$$

where $G(\mathbf{r};\mathbf{r}')$ is the Green's function associated with the operator $L(\mathbf{r})$ it is a two-point function (defined for points $\mathbf{r}$ and $\mathbf{r}'$) which satisfies the relationship

$$L(\mathbf{r})G(\mathbf{r};\mathbf{r}') = \delta(\mathbf{r}-\mathbf{r}'). \tag{7.38}$$

Now the original equation can be solved directly in terms of the Green's function by substituting Equation 7.37 into Equation 7.36

$$u(\mathbf{r}) = \int_{\Omega} G(\mathbf{r};\mathbf{r}')f(\mathbf{r}')d\mathbf{r}'. \tag{7.39}$$

The following section will derive an estimator for the quantity U_d at the centroid of the element e using a Green's function and the previously numerically calculated values of U_d at the nodes of the element.

2D Error Estimation

To formulate the error estimator note that that the differential operator (from Equation 3.28) for the diffusion equation is

$$L = \left[\nabla \cdot \nabla - \frac{\lambda}{\beta}\right]. \tag{7.40}$$

Thus, from Equation 7.38

$$(\nabla \cdot \nabla - \frac{\lambda}{\beta})G(\mathbf{r};\mathbf{r}') = \delta(\mathbf{r} - \mathbf{r}'). \tag{7.41}$$

This allows the following identity for the light diffusion equation to be defined:

$$\int_\Omega U_d(\mathbf{r})(\nabla \cdot \nabla - \frac{\lambda}{\beta})G(\mathbf{r};\mathbf{r}')d\Omega - \int_\Omega G(\mathbf{r};\mathbf{r}')(\nabla \cdot \nabla - \frac{\lambda}{\beta})U_d(\mathbf{r})d\Omega = 0. \tag{7.42}$$

Using Green's identities, the area integrals are converted into line integrals:

$$\begin{aligned}\int_{\Gamma_s} U_d(\mathbf{r})\ \hat{\mathbf{n}} \cdot \nabla G(\mathbf{r};\mathbf{r}')\ d\Gamma - \int_\Omega \nabla U_d(\mathbf{r}) \cdot \nabla G(\mathbf{r};\mathbf{r}')\ d\Omega \\ + \int_{\Gamma_s} G(\mathbf{r};\mathbf{r}')\ \hat{\mathbf{n}} \cdot \nabla U_d(\mathbf{r})\ d\Gamma - \int_\Omega \nabla G(\mathbf{r};\mathbf{r}') \cdot \nabla U_d(\mathbf{r})\ d\Omega = 0.\end{aligned} \tag{7.43}$$

The second and fourth integrals cancel resulting in

$$\int_{\Gamma_s} U_d(\mathbf{r})\ \hat{\mathbf{n}} \cdot \nabla G(\mathbf{r};\mathbf{r}')\ d\Gamma + \int_{\Gamma_s} G(\mathbf{r};\mathbf{r}')\ \hat{\mathbf{n}} \cdot \nabla U_d(\mathbf{r})\ d\Gamma = 0. \tag{7.44}$$

Considering again Equation 7.42, note that the term $(\nabla \cdot \nabla - \lambda/\beta)G(\mathbf{r};\mathbf{r}')$ in the first integral can be replaced by the delta-function from Equation 7.41 while the term $(\nabla \cdot \nabla - \lambda/\beta)U_d(\mathbf{r})$ in the second integral can be replaced with the term $-S(\mathbf{r})/\beta$ from Equation 3.28. Thus Equation 7.42 can be rewritten as

$$U_d(\mathbf{r}') - \int_\Omega G(\mathbf{r};\mathbf{r}')\frac{S(\mathbf{r})}{\beta}d\Omega = 0. \tag{7.45}$$

Equating Equation 7.45 and Equation 7.42 and simplifying yields

$$U_d(\mathbf{r}') = \int_{\Gamma_s} U_d(\mathbf{r})\ \hat{\mathbf{n}}\cdot\nabla G(\mathbf{r};\mathbf{r}')\ d\Gamma + \int_{\Gamma_s} G(\mathbf{r};\mathbf{r}')\ \hat{\mathbf{n}}\cdot\nabla U_d(\mathbf{r})\ d\Gamma + \int_\Omega G(\mathbf{r};\mathbf{r}')\frac{S(\mathbf{r})}{\beta}d\Omega, \tag{7.46}$$

where the Green's function $G(\mathbf{r};\mathbf{r}')$ is

$$G(\mathbf{r};\mathbf{r}') = \frac{1}{4j}H_0^{(2)}\left(-j\frac{\lambda}{\beta}|\mathbf{r}-\mathbf{r}'|\right) \tag{7.47}$$

and $H_0^{(2)}$ is a Hankel function. Equation 7.47 can be rewritten in terms of a modified Bessel function using the following identity [1]

$$\begin{aligned} K_\nu(z) &= -\frac{1}{2}\pi j e^{-\frac{1}{2}\nu\pi j}H_\nu^{(2)}\left(ze^{-\frac{1}{2}\pi j}\right) && (7.48)\\ K_0(z) &= -\frac{1}{2}\pi j H_0^{(2)}\left(ze^{-j\frac{\pi}{2}}\right) && (7.49)\\ 2\pi j K_0\left(ze^{\frac{1}{2}\pi j}\right) &= H_0^{(2)}(z). && (7.50) \end{aligned}$$

Substituting the Hankel function from Equation 7.50 into Equation 7.47 and simplifying yields

$$\begin{aligned} G(\mathbf{r};\mathbf{r}') &= \frac{1}{4j}2\pi j K_0\left(-j\frac{\lambda}{\beta}|\mathbf{r}-\mathbf{r}'|e^{j\frac{\pi}{2}}\right) && (7.51)\\ &= \frac{\pi}{2}K_0\left(\frac{\lambda}{\beta}|\mathbf{r}-\mathbf{r}'|\right). && (7.52) \end{aligned}$$

Equation 7.46 can be used as an error estimator if the calculated values for $U_d(\mathbf{r})$ are used on the right hand side, then compared to the interpolated solution at $\mathbf{r}'$ on the left with the result of Equation 7.46.

Evaluation of Terms in Equation 7.46 The terms in Equation 7.46 are not trivial to calculate. Each term and its numerical or analytical solution is discussed below.

First Term:

This term is measuring the error associated with the linear interpolation functions along the boundary. If U_d were known across the entire boundary this term would be completely accurate. However, U_d is only known at the nodes, thus during the evaluation of this integral the solution must be interpolated. Thus, the less accurate the interpolation, the more error occurs.

$$\int_{\Gamma_s} U_d(\mathbf{r})\ \hat{\mathbf{n}} \cdot \nabla G(\mathbf{r};\mathbf{r}')\ d\Gamma \tag{7.53}$$

$$= \int_{\Gamma_s} U_d(\mathbf{r}) \frac{\pi}{2} \hat{\mathbf{n}} \cdot \nabla K_0 \left(\frac{\lambda}{\beta} |\mathbf{r} - \mathbf{r}'| \right) d\Gamma \tag{7.54}$$

$$= \int_{\Gamma_s} U_d(\mathbf{r}) \frac{\pi}{2} \hat{\mathbf{n}} \cdot \nabla K_0 \left(\frac{\lambda}{\beta} \sqrt{(x - x')^2 + (y - y')^2} \right) d\Gamma \tag{7.55}$$

where x and y are the x and y coordinates of $\mathbf{r}$ and x' and y' are the x and y coordinates of $\mathbf{r}'$. Let $u = \lambda\sqrt{(x - x')^2 + (y - y')^2}/\beta$ then

$$\int_{\Gamma_s} U_d(\mathbf{r})\hat{\mathbf{n}}\cdot\nabla G(\mathbf{r};\mathbf{r}')d\Gamma \tag{7.56}$$

$$= \int_{\Gamma_s} U_d(\mathbf{r})\frac{\pi}{2}\,\hat{\mathbf{n}}\cdot\left[\frac{\partial K_0(u)}{\partial u}\left(\frac{\partial u}{\partial x}\hat{\mathbf{x}}+\frac{\partial u}{\partial y}\hat{\mathbf{y}}\right)\right]d\Gamma \tag{7.57}$$

$$= \int_{\Gamma_s} U_d(\mathbf{r})\frac{\pi}{2}\,\hat{\mathbf{n}}\cdot\left[K_1(u)\left(\frac{\lambda}{\beta}\frac{(x-x')\hat{\mathbf{x}}+(y-y')\hat{\mathbf{y}}}{\sqrt{(x-x')^2+(y-y')^2}}\right)\right]d\Gamma \tag{7.58}$$

$$= \int_{\Gamma_s} U_d(\mathbf{r})\frac{\pi\lambda K_1(\lambda|\mathbf{r}-\mathbf{r}'|/\beta)}{2\beta|\mathbf{r}-\mathbf{r}'|}\left[(x-x')|\hat{\mathbf{n}}_{\mathbf{x}}|+(y-y')|\hat{\mathbf{n}}_{\mathbf{y}}|\right]d\Gamma. \tag{7.59}$$

Equation 7.59 can then replace the first integral in Equation 7.51 and solved numerically.

Second Term:

This term measures the "jump discontinuity" which occurs with the flux between elements. Mathematically, the flux is constant across the boundary, in fact this weak boundary condition was used in the derivation of the finite element formulation. In practice however, the interpolation functions do not smoothly transition from one element to the other. The evaluation of this term measures the error from this discontinuity. Thus,

$$\int_{\Gamma_s} G(\mathbf{r};\mathbf{r}')\hat{\mathbf{n}}\cdot\nabla U_d(\mathbf{r})d\Gamma \tag{7.60}$$

$$= \int_{\Gamma_s} G(\mathbf{r};\mathbf{r}')\hat{\mathbf{n}}\cdot\nabla\left(\sum_{j=1}^{3}\phi_j^e(\mathbf{r})U_{d_j}^e\right)d\Gamma \tag{7.61}$$

$$= \int_{\Gamma_s} G(\mathbf{r};\mathbf{r}')\left(\sum_{j=1}^{3}\left(\frac{b_i^e}{2\Delta_e}|\hat{\mathbf{n}}_x|+\frac{c_i^e}{2\Delta_e}|\hat{\mathbf{n}}_y|\right)U_{d_j}^e\right)d\Gamma \tag{7.62}$$

$$= \int_{\Gamma_s} K_0\left(\lambda|\mathbf{r}-\mathbf{r}'|/\beta\right)\frac{\pi}{2}\left(\sum_{j=1}^{3}\left(\frac{b_i^e}{2\Delta_e}|\hat{\mathbf{n}}_x|+\frac{c_i^e}{2\Delta_e}|\hat{\mathbf{n}}_y|\right)U_{d_j}^e\right)d\Gamma \tag{7.63}$$

The integral along the boundary can now be computed analytically. Note that the normal terms ($\hat{\mathbf{n}}_d$) are the normals of the current boundary. Also, when evaluating the boundary integral it is not clear which element should contribute its elemental constants, (b_i^e, c_j^e, and Δ_e). Choosing the element which contains the point $\mathbf{r}'$ does not accurately reflect the error associated with the jump discontinuity since the interpolation functions will be the same ones used to calculate U_d in the first place. Instead choosing the element constants to be the neighboring element's constants will accurately reflect the jump discontinuity error in this case.

Third Term:

This term is another measure of the error associated with the interpolation functions. Specifically in this case the interpolation error associated with the source term. As seen in the derivation of the transport equation (Equation 3.2), the reduced incident intensity falls off exponentially as it traverses through the volume. This term evaluates the inaccuracy of the linear interpolation of the source by integrating the source across the entire element. Unfortunately there is no closed form for the source term, thus to maintain a higher accuracy in the error calculation a higher order Gaussian quadrature method is used in the final evaluation of the integral.

$$\int_\Omega G(\mathbf{r};\mathbf{r}')\frac{S(\mathbf{r})}{\beta}d\Omega \tag{7.64}$$

$$= \int_\Omega \frac{\pi}{2}K_0\left(\frac{\lambda}{\beta}|\mathbf{r}-\mathbf{r}'|\right)\frac{S(\mathbf{r})}{\beta}d\Omega \tag{7.65}$$

This function is problematic to integrate since a singularity exists in the Green's function at $\mathbf{r}=\mathbf{r}'$. Although the singularity can be removed, the resulting asymptotic function is just as problematic to calculate.

However, the author's experimental results have shown that the integrand in Equation 7.64 is quite smooth except very close to its singularity. To verify this, note that the the asymptotic behavior of $K_0(z)$ for small values ($0 < z \ll 1$) is [1],

$$K_0(z) \rightarrow -\ln(z/2) - \gamma \tag{7.66}$$

where γ is the Euler-Mascheroni constant. As an experiment one can construct a two dimensional element such that the entire side lies in a singularity as the following function.

$$f(x, y) = ln(x) \tag{7.67}$$

Given a right triangle with coordinates (0,0), (d,d), (d,0) one can analytically integrate the above function:

$$\begin{aligned} \int_0^d \int_0^y \ln x \; dx \; dy &= \int_0^d y \ln y - y \; dy & (7.68) \\ &= \frac{y^2}{2}\left(\ln y - \frac{3}{2}\right)\Big|_0^d & (7.69) \\ &= -\frac{d^2}{2}\left(\ln d - \frac{3}{2}\right) & (7.70) \end{aligned}$$

Using a adaptive method with 12-point 6th order method (from [13]) which has no abscissa located on the singularity. The accurate results in Table 7.1 were achieved by using an adaptive Gaussian quadrature routine based on a 6 and 12 point triangular Gaussian quadrature which has no points lying on the centroid or boundaries. Since the modified Bessel function behaves likes a log near its singularity, this is proof that Gaussian quadrature methods can easily handle such a singularity.

d	**Analytic Result**	**Numerical Result**
5.000000e-01	-2.741434e-01	-2.741366e-01
2.500000e-01	-9.019670e-02	-9.019329e-02
1.250000e-01	-2.796439e-02	-2.796268e-02
6.250000e-02	-8.344900e-03	-8.344047e-03
3.125000e-02	-2.424676e-03	-2.424250e-03
1.562500e-02	-6.907816e-04	-6.905690e-04
7.812500e-03	-1.938486e-04	-1.937426e-04
3.906250e-03	-5.375044e-05	-5.369784e-05
1.953125e-03	-1.475968e-05	-1.473376e-05
9.765625e-04	-4.020439e-06	-4.007847e-06

Table 7.1: Results of Gaussian quadrature routine on a simple triangle with a modified Bessel function singularity. These results show that adaptive Gaussian quadrature is a feasible method for directly evaluating the function even though the singularity still resides in it.

Boundary Conditions In the case of a non boundary element the error estimation equation is

$$U_d(\mathbf{r}') = \int_{\Gamma_s} U_d(\mathbf{r})\ \hat{\mathbf{n}} \cdot \nabla G(\mathbf{r};\mathbf{r}')\ d\Gamma + \int_{\Gamma_s} G(\mathbf{r};\mathbf{r}')\ \hat{\mathbf{n}} \cdot \nabla U_d(\mathbf{r})\ d\Gamma + \int_{\Omega} G(\mathbf{r};\mathbf{r}')\frac{S(\mathbf{r})}{\beta} d\Omega. \tag{7.71}$$

However, in the presence of a boundary the first and second integrals should be replaced with the boundary condition

$$\mu_{tr} U_d(\mathbf{r}) - \frac{2}{3}\frac{\partial}{\partial n^-} U_d(\mathbf{r}) + \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi} = 0. \tag{7.72}$$

Then Equation 7.71 becomes:

$$
\begin{aligned}
U_d(\mathbf{r}') &= \int_{\Gamma_s} \frac{1}{\mu_{tr}} \left(\frac{2}{3} \hat{\mathbf{n}} \cdot \nabla U_d(\mathbf{r}) - \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi} \right) \hat{\mathbf{n}} \cdot \nabla G(\mathbf{r};\mathbf{r}')\, d\Gamma \\
&\quad + \int_{\Gamma_s} G(\mathbf{r};\mathbf{r}')\, \frac{3}{2} \left(\mu_{tr} U_d(\mathbf{r}) + \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi} \right) d\Gamma \\
&\quad + \int_{\Omega} G(\mathbf{r};\mathbf{r}') \frac{S(\mathbf{r})}{\beta} d\Omega.
\end{aligned} \tag{7.73}
$$

The expansion of the $\hat{\mathbf{n}} \cdot \nabla G(\mathbf{r};\mathbf{r}')$ and $\hat{\mathbf{n}} \cdot \nabla U_d(\mathbf{r})$ terms have been described in Equation 7.59 and 7.63. Replacing these terms and as well as formally defining the 2D Green's function yields

$$
\begin{aligned}
U_d(\mathbf{r}') &= \int_{\Gamma_s} \frac{1}{\mu_{tr}} \left(\frac{2}{3} \left(\sum_{j=1}^{3} \left(\frac{b_i^e}{2\Delta_e} |\hat{\mathbf{n}}_x| + \frac{c_i^e}{2\Delta_e} |\hat{\mathbf{n}}_y| \right) U_{d_j}^e \right) - \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi} \right) \\
&\quad \cdot \frac{\pi\lambda K_1(\lambda|\mathbf{r}-\mathbf{r}'|/\beta)}{2\beta|\mathbf{r}-\mathbf{r}'|} \left[(x-x')|\hat{\mathbf{n}}_{\mathbf{x}}| + (y-y')|\hat{\mathbf{n}}_{\mathbf{y}}| \right] d\Gamma \\
&\quad + \int_{\Gamma_s} \frac{\pi}{2} K_0 \left(\frac{\lambda}{\beta} |\mathbf{r}-\mathbf{r}'| \right) \frac{3}{2} \left(\mu_{tr} U_d(\mathbf{r}) + \frac{2\hat{\mathbf{n}}^- \cdot \mathbf{Q}_1(\mathbf{r})}{4\pi} \right) d\Gamma \\
&\quad + \int_{\Omega} K_0 \left(\frac{\lambda}{\beta} |\mathbf{r}-\mathbf{r}'| \right) \frac{S(\mathbf{r})}{\beta} d\Omega.
\end{aligned} \tag{7.74}
$$

Note however, that these formulations should only be used on those regions which lie on the boundaries.

7.3 Results

The Green's function *a posteriori* error estimator presented in this chapter is able to accurately predict and converge on an error for elements even in the presence of a source. Figure 7.3 shows the error progression for the centroid of the element shown in Figure 7.1 which is successively subdivided. Note that when the element is large the estimator does not accurately predict the error, but once the element is "small enough" the error converges rapidly.

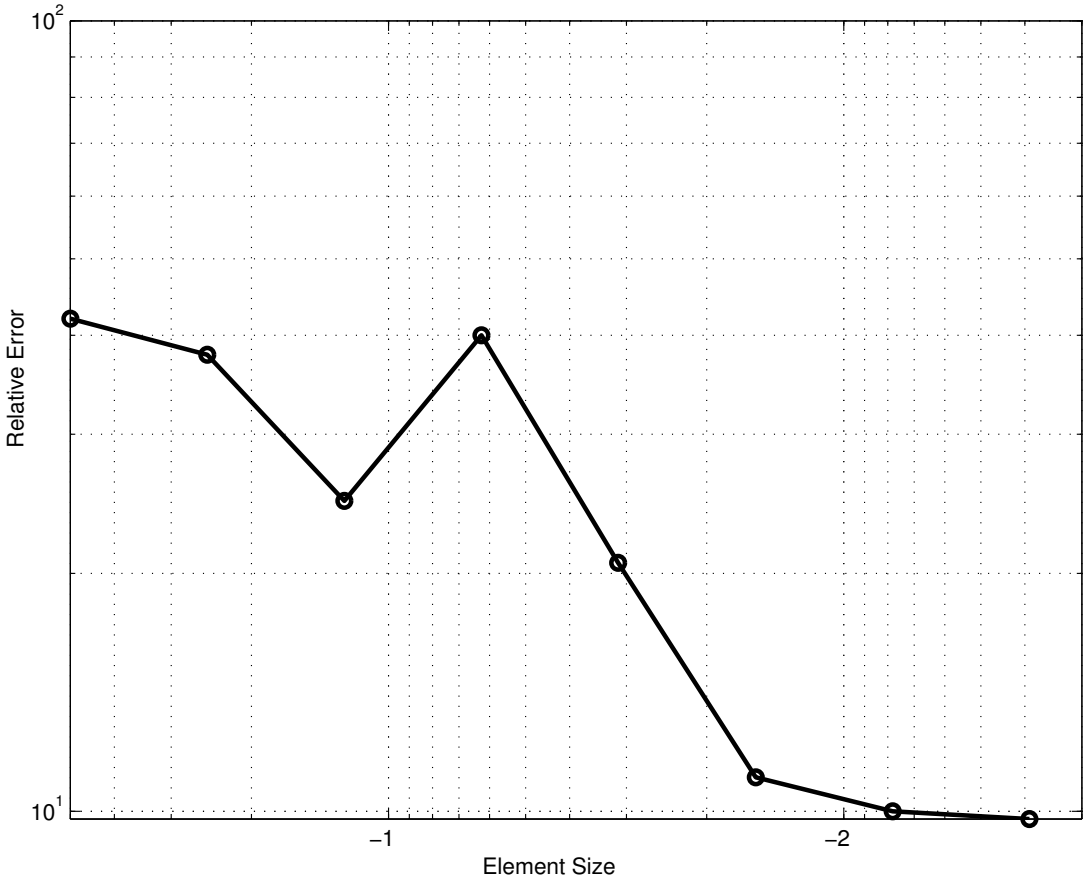


Figure 7.3: This graph shows the progression of the error estimation rate for the Green's function *a posteriori* estimator in the presence of a source for the element shown in Figure 7.1. Notice for large elements the estimator is somewhat unstable, but for smaller element sizes the estimator converges. The error in this graph is the ratio between the numerical value of $U_{ri}(\mathbf{r}')$ and the estimated value of $U_{ri}(\mathbf{r}')$ calculated by the Green's function formulation. The "element size" axis is the length in cm of the edge of the element (an equilateral triangle).

7.4 Conclusions

The work presented in this chapter derives a self-equilibrated residual and complementary *a posteriori* error estimator for finite differences and demonstrated that it was not suitable for practical error estimation. Furthermore, an alternative method of moments estimator which uses a Green's function to solve the average diffuse intensity was proposed. Experimentally, this method accurately predicts the relative error associated with elements in the finite element solution.

CHAPTER 8

PHYSICS-BASED SUBSURFACE VISUALIZATION OF HUMAN TISSUE

"What a dog I got, his favorite bone is in my arm."

— *Rodney Dangerfield.*

This chapter contains a description for an optical and biological model for volume visualization which mimics the optical scattering in human tissue seen by near-infrared (NIR) transillumination. Figure 8.1 illustrates a real-world transillumination device which is useful for determining the subsurface location of veins or arteries for vascular taps. In recent years researchers have leveraged the property of human tissue NIR[22] permeability to develop methods for non-invasive imaging and detection to locate tumors, determine blood oxygenation levels, assess nutritional absorption, brain activity, water content, and many others [8, 21, 25, 29, 37, 58, 71]. The model presented in this chapter is an implementation of the three dimensional finite element method presented in Chapter 5 to simulate the physical process of light scattering for the NIR transillumination configuration.

There is a need for viable optical models and computational methods that study light transport through biological tissue at a variety of scales and wavelengths. Most

[22]The spectral region of NIR light is considered to be in the wavelength range of 700 to 5000 nm.

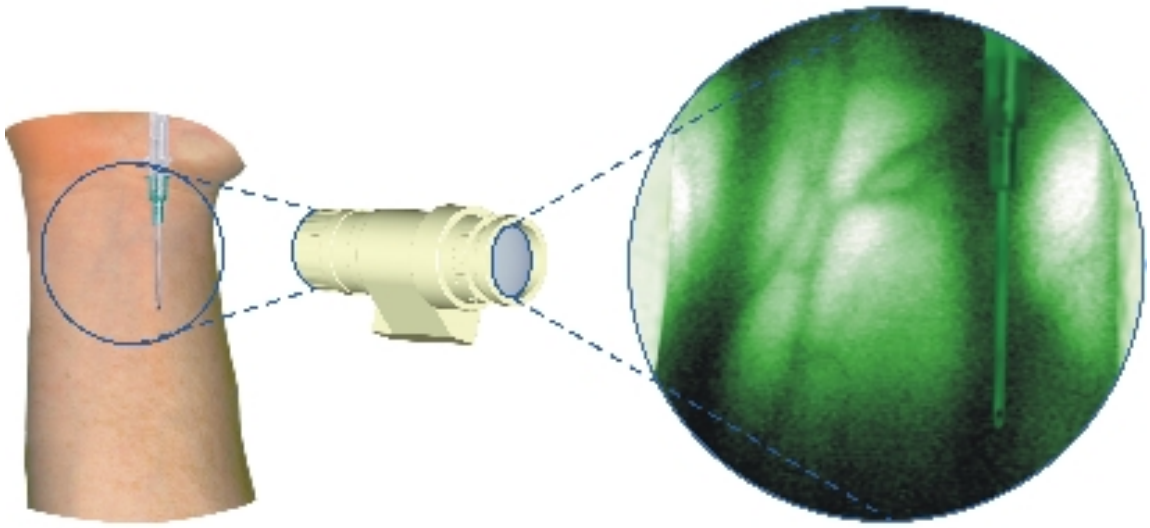


Figure 8.1: This figure illustrates the use of the Vascular Viewer^{TM} which illuminates the far side of the forearm with an array of infrared LEDs, shines the light through the arm, then is viewed directly using a filtered infrared scope.

existing optical models [47] and volume rendering methods based on those do not display internal structure in a realistic manner since they do not simulate the high amount of scattering that occurs. However, in computer graphics literature, subsurface scattering methods have been employed to increase the realism of organic surfaces and human skin [34]. Unfortunately, these methods, are not physically accurate and cannot be used for the purposes described below.

8.1 Biological and Geometrical Phantom

The geometry for the biological model presented in this chapter (a human forearm) was created in the finite element meshing software, ANSYS. The model (shown in Figure 8.2) includes both the ulnar and radial artery, superficial veins, the radius and

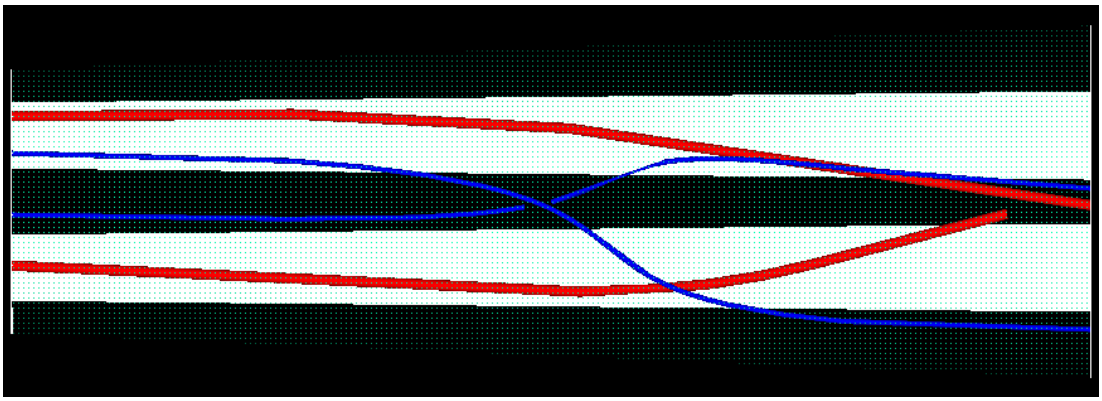


Figure 8.2: ANSYS model for human forearm. Red vessels are arteries and blue vessels are superficial veins. Inner cylinders represent the radius and ulna bones.

ulna bones, and an outer "muscle" layer. Note that the ulnar and radial arteries are those which show up in the transillumination photograph in Figure 8.4b.

Although it is difficult to measure *in vivo* scattering values for internal human tissues many studies exist which measure the absorption coefficient μ_a and the reduced scattering coefficient μ_s' in a variety of tissues at different wavelengths [9, 12, 20, 45]. The scattering properties chosen by the author in Table 8.1 were for a wavelength of 950nm.

Tissue	μ_s (cm^{-1})	μ_a (cm^{-1})	g
Bone [19]	240	0.5	0.945
Muscle [63]	55.5†	0.456	0.9
Blood [12]*	2.84	505	0.992

Table 8.1: Optical properties for the simulation shown in Figures 8.4a and 8.5. The source wavelength is assumed to be 950nm. (†Assuming $g = 0.9$.; *Wavelength = 960nm.)

8.1.1 Optical Properties of Skin

NIR light is absorbed in the dermis mainly by water and collagen [68]. Although skin thickness decreases with age [66] the water content of the skin is greater with increasing age when expressed per unit weight [44]. Furthermore the proportion of insoluble collagen increases with age while that of soluble collagen decreases [26].

The total skin thickness is quite small compared to the entire volume: in young men the total thickness is 1.0-1.2mm and young women 0.8-1.0mm and decreases to 0.7-0.9mm in both by age 70 [66]. Since the skin layer is relatively thin compared to the volume of the arm, its major effect on the light propagation occurs as light enters and leaves the medium. Since the boundary condition assumes no inward directed diffuse intensity, the absorbing boundary layer will not have an effect on the diffuse intensity computations. This occurs since intensity cannot be reflected from the boundary, absorbed, then reflected back into the medium. Thus it is accurate to model the effect of NIR absorption by the skin layer by scaling the reduced incident intensity by a value proportional to the rate of absorption. This is essentially a modification of the particular solution to Equation 3.1 such that it scales the initial intensity by some percentage:

$$I_{ri}(\mathbf{r}, \hat{\mathbf{s}}) = \Lambda(t) I_{ri}(\mathbf{r}_0, \hat{\mathbf{s}})\ e^{-\int \mu_t(\mathbf{s})\mathbf{s}\ d\mathbf{s}} \tag{8.1}$$

where $\mathbf{r}_0$ is the location of the light source and $\Lambda(t)$ is a parameter based percentage between $[0, 1]$ which accounts for the NIR absorption in the skin. This term can be calculated by exponentially decaying the absorption coefficient for skin [29] across the depth of the skin

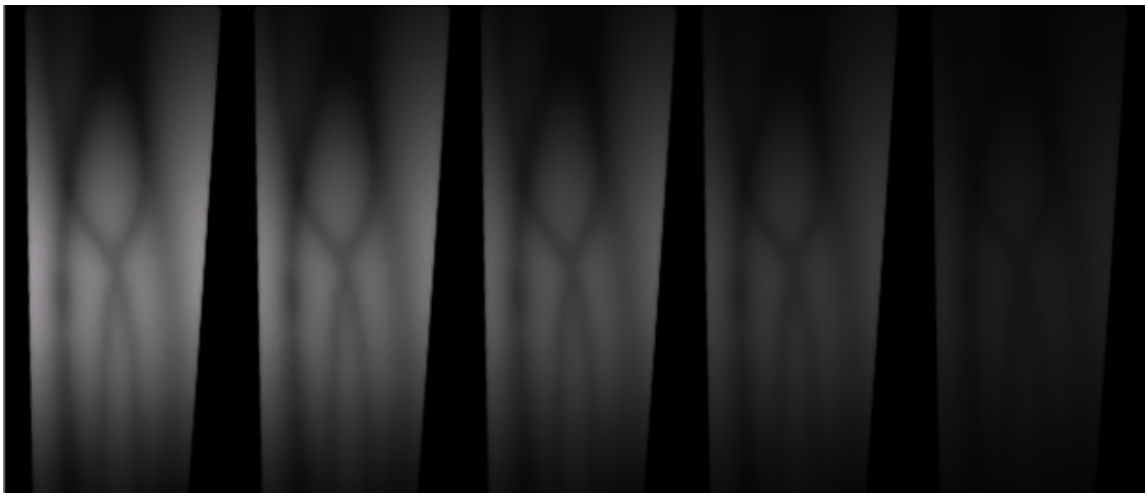


Figure 8.3: Renderings of the arm dataset as the skin thickness is increased. The left image allows 100% of the NIR light into and out of the model while the right image reduces the NIR penetration to only 20% as scaled by the $\Lambda(t)$ parameter in Equation 8.2.

$$\Lambda(t) = e^{-\delta(t)\mu_a(t)} \tag{8.2}$$

where $\delta(t)$ is the thickness of the skin and $\mu_a(t)$ is the absorption cross section for that section of skin.

Likewise, as $U_d(\mathbf{r})$ and $U_{ri}(\mathbf{r})$ are projected to the detector the output intensity is scaled by $\Lambda(t)$ to account for the absorption of light exiting the skin.

8.2 Implementation

The implementation of this model does not differ greatly from the 2D described earlier in Section 6.2. The construction of the finite element matrix $[K]$ is identical except the 3D formulation developed in Chapter 5 is used instead. For simplicity, the source calculation was changed slightly to use an octree to accelerate ray intersection tests.

8.2.1 Source Distribution $\{b\}$

The source distribution is calculated from solving for $U_{ri}(\mathbf{r})$ at every node in the mesh (see Equation B.2) then integrating across the elements as shown in Equation 5.104. Calculating U_{ri} involves tracing a ray from the node to the light source and attenuating the source light based on the extinction cross section (μ_t) of the elements the ray passes through. Since it is possible to have different material properties in each element of the mesh we must calculate an intersection for each element the ray passes through, as opposed to the straight line distance from the node to the edge of the mesh. To avoid expensive intersection tests with all the elements in the node, the domain containing all elements of the FE grid is partitioned into an octree. The intersection task is then simplified by testing only those elements that reside in octants that intersect the ray. For a well balanced octree this reduces the number of potential intersection checks from linear to logarithmic complexity. The algorithm for building the source vector $\{b\}$ is shown in Algorithm 8.2.1.

8.2.2 Rendering

To render the simulated NIR images in this chapter, $U_d(\mathbf{r})$ is first solved for at all nodes in the mesh then interpolated onto a complete volume. If the nodal solutions are projected directly to the surface of the model the solution tends to look aliased. Instead, the solved volume is loaded as a 3D volume of scalar intensities into Kitware's VolView volume visualization software [30]. Allowing a bit of translucency in the volume generates much more realistic renderings. Volume rendering the solution is essentially similar to modeling the subtle forward scattering effects as light diffuses out of the material which are not completely modeled by the diffusion equation.

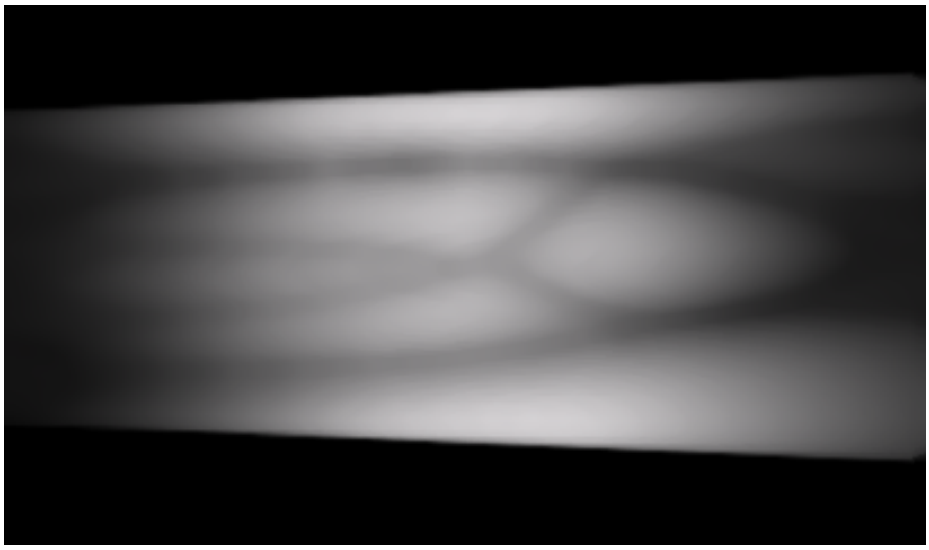
Algorithm 1 Algorithm to build $\{b\}$.

`Build_w`($\{b\}$):
  for each node n in the mesh **do**
    Trace a ray from the source through the medium to n
      and calculate $U_{ri}(\mathbf{r})$based on Equation 3.22
    Store this result in `uri(n)`
  end for
  for each element e in the mesh **do**
    Build a local 4 element vector $\{b^e\}$ given Equation 5.104 and
      using the precomputed values for $U_{ri}(\mathbf{r})$from `uri(n)`
    Add boundary conditions from Equation 5.113 to $\{b^e\}$
    Add the elements from $\{b^e\}$ to $\{b\}$ by mapping local node
      numbers to global ones as in Equation 5.57
  end for
end *//Build_w*

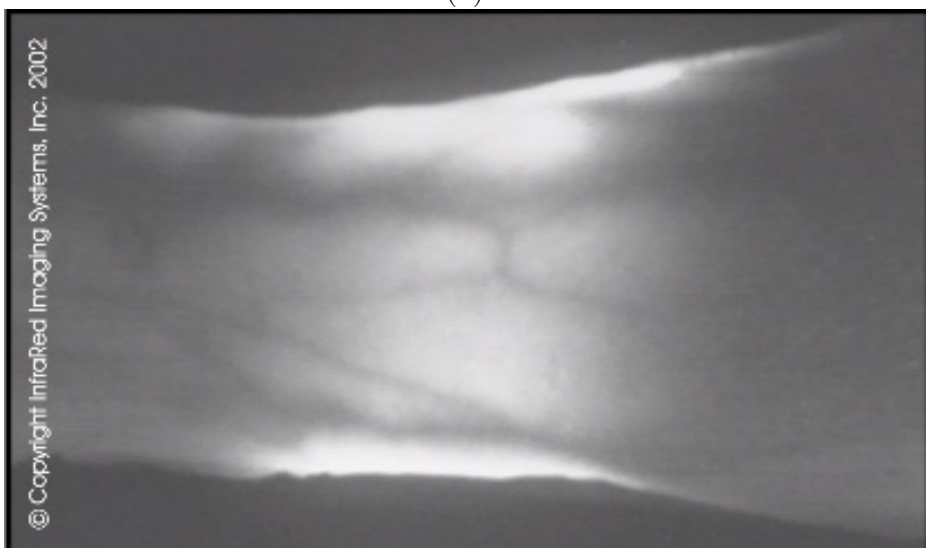
8.3 Results

A primary advantage of this method is the ability to perform simulations to determine maximum visible depths. One experiment the author performed was to transilluminate a discontinuity which could be moved below the surface. As the discontinuity is moved the mean intensity increases due to more light being allowed to reach that area on the surface. Simultaneously, the standard deviation of the intensity histogram decreases, reflecting the fact that the intensity is distributed more evenly across the surface. The images in Figure 8.6 demonstrate the visual aspects of this experiment while the plot in Figure 8.7 shows the visibility metric as the discontinuity moves beneath the surface. For perspective and to illustrate the effect of source placement Figure 8.5 shows a rendering of the arm with both a top and bottom lit configuration.

Some sample run times and matrix fill statistics are given in Table 8.2. As mentioned previously, the "Factor + $[K]$ Creation Time" column can be thought of as



(a)



(b)

Figure 8.4: Figure (a) shows the simulation of near infrared light through a sample arm model while Figure (b) shows the actual result from the Vascular Viewer^{TM}. Note that the Figure (b) not only shows the NIR light scattered through the arm but also any background light which happens to be in the environment, the simulation in Figure (a) lacks this background NIR noise.

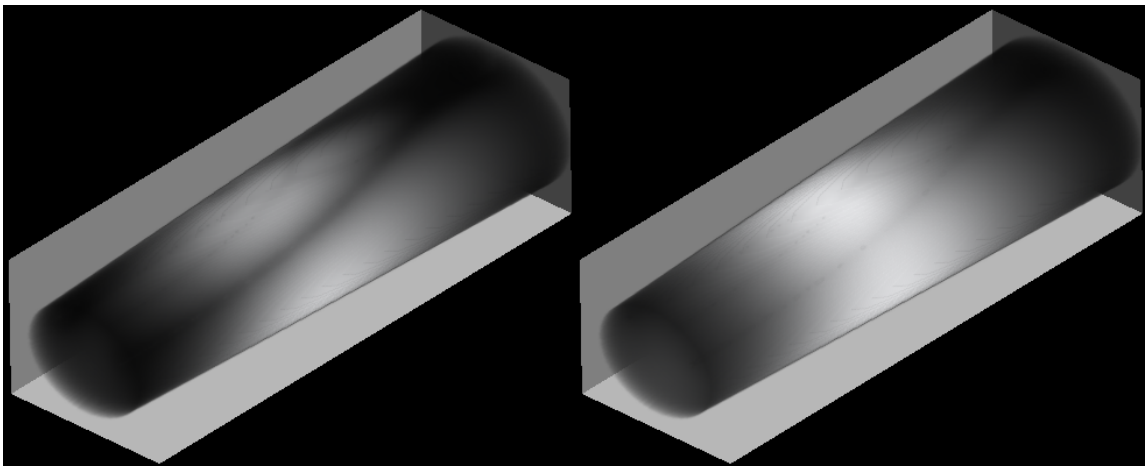


Figure 8.5: This figure shows the solution of $U_d(\mathbf{r})$ on the finite element model shown in Figure 8.2. The arm on the left is lit from the bottom while the image on the right is lit from the top. The planes in the background are shown for perspective.

a precomputation step if multiple simulations are run with a moving light source. Saving the factorization reduces successive calculations by approximately half.

8.4 Conclusions

The content of this chapter covers a practical framework for solving light diffusion problems in three dimensions for the purpose of simulating light scattering in inhomogeneous materials. Specifically, it has been shown that a finite element solution to the diffusion equation is able to accurately capture many subsurface scattering effects seen in real world NIR transillumination.

# of Elements	Non-zeros in $[K]$	Non-zeros in $[L]$	Source Creation Time	Factor + $[K]$ Creation Time	Solution Time
5000	8261	4.30e4	1.187s	2.141s	0.046s
16875	26416	3.20e5	5.156s	7.266s	0.172s
40000	60921	8.09e5	17.172s	17.22s	0.562s
78125	117026	2.65e6	37.937s	34.204s	1.563s

Table 8.2: Progression for timings and non-zero fill rates as 3D mesh size is increased. These results were generated from a 1cm^2 cube which was evenly meshed. The solution time is the amount of time to calculate $\{U_d\}$ from $[K]\{U_d\} = \{b\}$ given the factorization $[L]$.

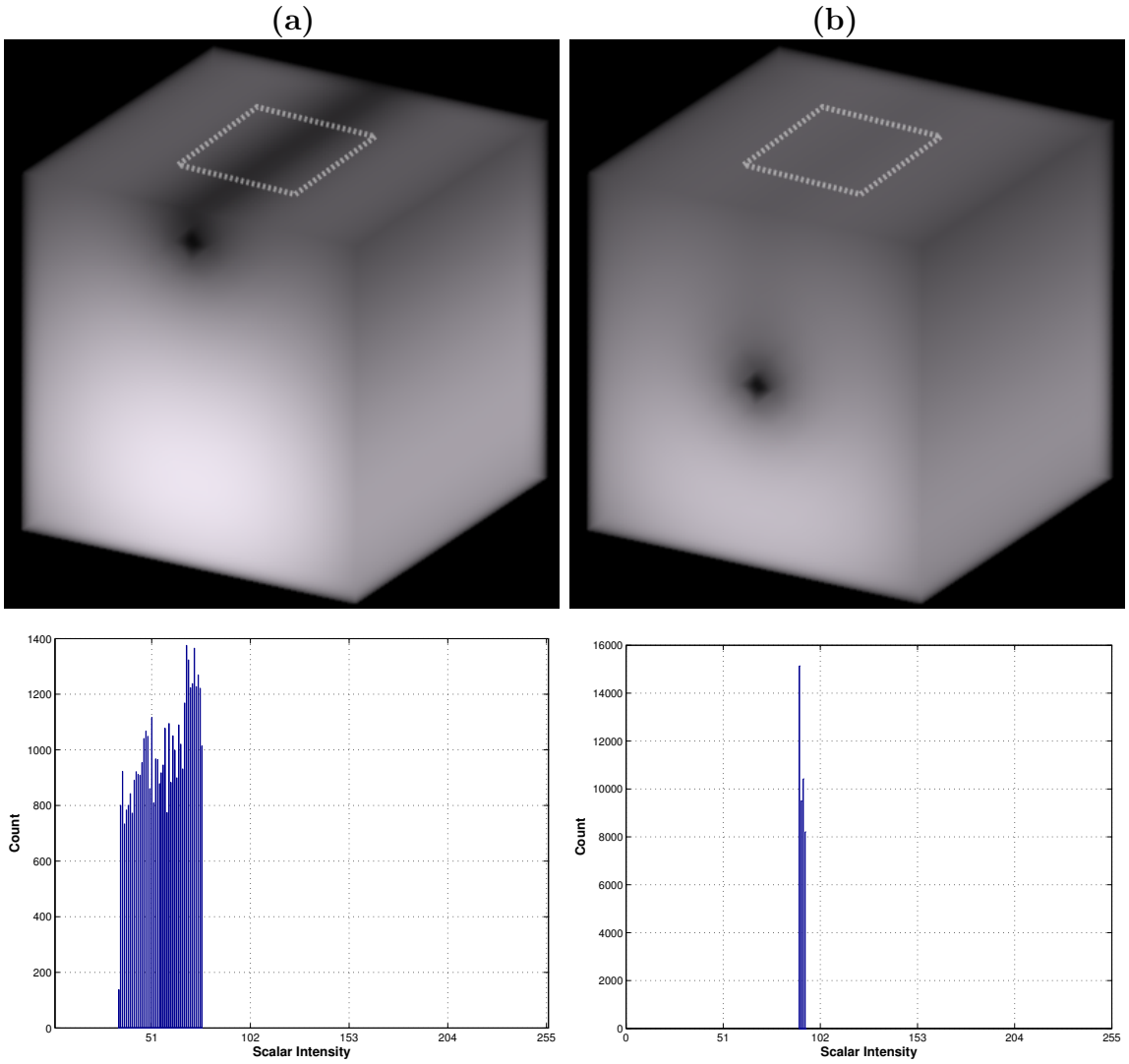


Figure 8.6: This figure illustrates the metric for determining visibility of a subsurface structure in a $10cm^3$ medium. Column **(a)** places the discontinuity at 1cm while column **(b)** places it at 5cm. The histogram plots on the bottom row show the intensity of the pixels around the white boxes drawn on the top surface of the cubes. Notice as the discontinuity sinks, the average intensity increases while the standard deviation of the histogram decreases.

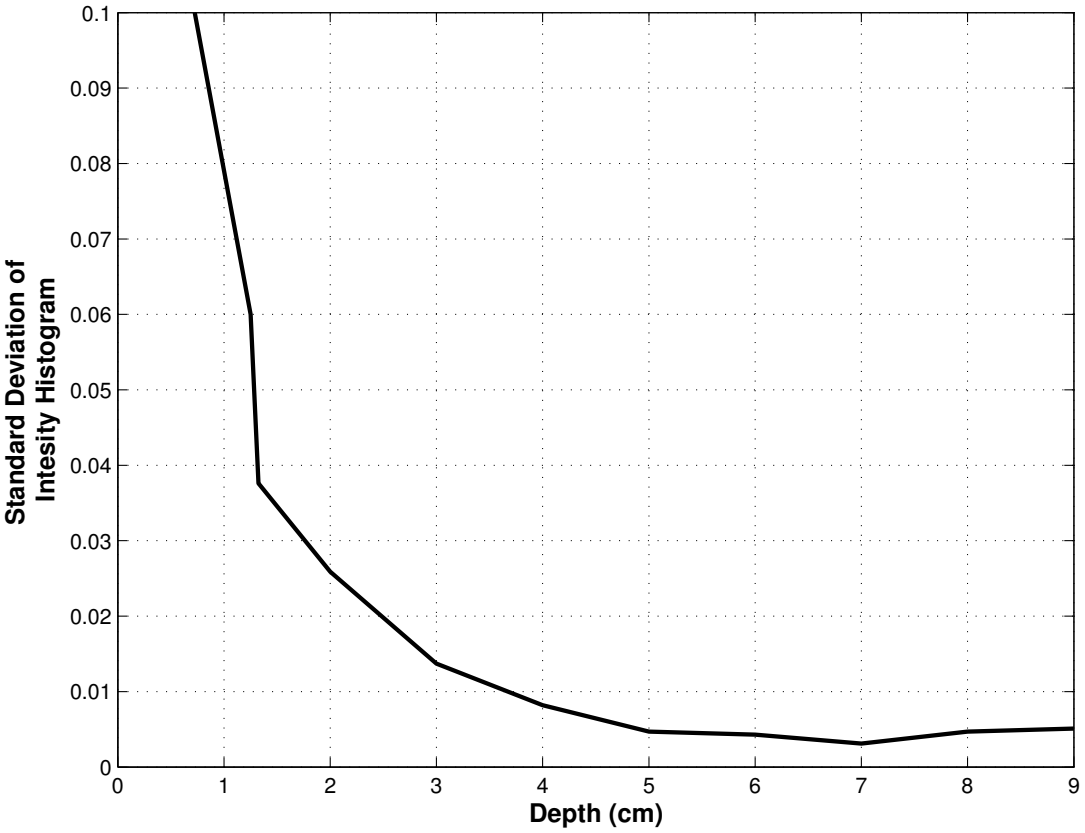


Figure 8.7: This figure shows how the standard deviation of intensity varies as the discontinuity in Figure 8.6 moves below the surface. $\mu_s = 10\text{cm}^{-1}$ $\mu_a = 0.1\text{cm}^{-1}$. The discontinuity visibility drops significantly after 1.5cm.

CHAPTER 9

CONCLUSIONS AND CONTINUING WORK

"*Mehr Licht!*"

— Purported last words of
Johann Wolfgang von Goethe.

9.1 Review

This dissertation presented the formal problem of calculating subsurface scattering in diffuse materials, specifically for near infrared light in human tissue. Furthermore, background research in computer graphics and other biomedical related work into subsurface scattering was presented in Chapter 2. The mathematical background for the transport equation and diffusion approximation was presented in Chapter 4. Then, a proposed solution to the diffusion equation in terms of finite differences was presented was presented in Chapter 4. Although the finite difference method generated good visual results, it was unsuitable for precise calculations on arbitrary geometries and complex scattering patterns. This lead to the presentation of the finite element solution to the diffusion equation in Chapter 5. The finite element approach adapted easily to arbitrary geometries and even adaptable meshes which can be refined in regions of high error. In Chapter 6 a 2D hanging node mesh refinement

method was presented along with an efficient implementation of the finite difference solution to the diffusion approximation. Chapter 7 continued with a presentation of a robust per-element method of moments error estimation scheme. Finally, an efficient full 3D implementation of the finite element diffusion scheme was presented in Chapter 8 and applied to and compared to a real world NIR visualization device.

9.2 Discussion

This dissertation provided techniques to answer the question "What is the complete distribution of light in arbitrarily shaped tissues with varying optical properties?" Specifically a finite difference and finite element approach to solving the light diffusion equation was presented. These techniques were demonstrated to accurately calculate the distribution of visible and NIR light in arbitrary geometries with varying inhomogeneous light scattering properties. It continues with providing a method for answering the question "How can the error in a finite element solution of light scattering be estimated with the diffusion approximation?" This is accomplished with a novel error estimation technique based on a Green's function estimator which can be used to predict the error in the finite element solution and refine those areas with high error.

Computer graphics models often skim over, or oversimplify the physics of a particular light phenomenon. This is particularly true in the case of subsurface scattering. Although results are often visually appealing, in the long run progress will be stunted if the scientific basis is diluted. On the other hand, the biomedical optics community is quite precise when it comes to the mathematical simulations, however often the implementations are lacking efficient frameworks. This is the main contribution of

this dissertation: to bring together both communities while contributing something to both. Specifically, to the computer graphics community it brings the physics of light diffusion to an practicable level as well as introducing relevant new problems to the community; such as the visualization of the NIR transillumination device. To the biomedical optics community it brings efficient implementations of previously known algorithms as well as introducing new techniques such as the hanging nodes refinement scheme, highlights the shortcomings of existing *a posteriori* error estimation schemes, and proposes a new error estimation scheme for adaptive refinement methods.

9.3 Continuing Work

There are several issues presented in this dissertation which can, and will, benefit from further examination.

9.3.1 Higher Dimensions

Although a 3D finite element implementation method was presented in Chapter 8 a 3D implementation of hanging nodes and a 3D error estimation scheme needs to be investigated. First, the 3D implementation of hanging nodes is not trivial. Tetrahedrons do not divide as simply as triangles do, and care must be taken during implementation to ensure that the data structure remains coherent. Secondly, the method of moments error estimator does not convert easily from 2D to 3D. At the very least the Green's kernel differs, thus changing the derivation and the implementation.

9.3.2 Real World Models

In Chapter 8 a synthetic numerical phantom was created using basic knowledge of human anatomy. It would be preferable to use a segmented MRI model however,

MRI systems are not configured to image human arms at high resolution. As part of the author's continuing work a plan is in place to configure an MR coil for optimal high resolution imaging of a human arm and compare images generated from the 3D diffusion simulation with images from the same arm visualized using the NIR transillumination device.

9.3.3 Computational Complexity

One difficulty with transitioning from a two dimensional model to a three dimensional one occurs with the amount of data required to store the model as well as the computational requirement of creating and solving the finite element matrix. Although the fill rates for Cholesky factorization are low for two dimensional models, the same is not true for three dimensions. As more complex models are simulated in three dimensions a robust sparse iterative solver will need to be implemented to process the solutions on such systems.

9.3.4 Other Uses of the Diffusion Model

It is possible to use the techniques and framework developed in this dissertation to simulate similar non-light diffusion problems. For example, radio frequency ablation (RFA), microwave ablation (MA), and cryoablation are techniques are used to treat a variety of diseases such as tumor removal from the liver, kidney or other organs; catheter ablation of heart arrhythmias; endometrial ablation as an alternative to hysterectomies; as well as many other conditions. During the ablation process a catheter is inserted into the tissue in question and a current, microwave signal, or coolant, is injected into the tissue. The hope is that the questionable tissue is destroyed by the extreme temperature change and healthy tissue is left alone. However, during

the procedure it is difficult for the physician to determine which tissues have been destroyed. A finite element simulation of the heat diffusion equation could provide such answers, and perhaps with MRI data from the patient, predict where a surgeon must apply the ablation treatment and at what intensity and duration.

9.4 Final Words

Out of all the techniques presented in this dissertation, this author has found that the finite element method is one of the most useful. Although this technique occasionally is used in computer graphics literature, it is often overlooked, perhaps because of its mathematical complexity. However, once one is familiar with the finite element formulation one can appreciate its power to solve equations over meshes of arbitrary orientation. Furthermore, the user has great sway over how to increase the order of accuracy of the solution through refining the mesh or increasing the order of interpolation functions. It is the hope of this author that such computational techniques will find their way into mainstream computer graphics research.

APPENDIX A

DERIVATION OF 2D AND 3D FINITE ELEMENT FORMULAE

This appendix provides the derivation for the formula used to analytically calculate the integration of linear interpolation functions across the elemental domain (Equations 5.69 and 5.96) as originally presented by Eisenberg and Malvern [17] with one error.

The equations in question are:

$$\iint_{\Omega^e} (\phi_i^e(\mathbf{r}))^l \left(\phi_j^e(\mathbf{r})\right)^m (\phi_k^e(\mathbf{r}))^n \; dx\, dy = \frac{l!\; m!\; n!}{(l+m+n+n+2)!} 2\Delta^e \tag{A.1}$$

and

$$\iiint_{V^e} (\phi_1^e(\mathbf{r}))^k (\phi_2^e(\mathbf{r}))^l (\phi_3^e(\mathbf{r}))^m (\phi_4^e(\mathbf{r}))^n \; dx\, dy = \frac{k!\; l!\; m!\; n!}{(k+l+m+n+n+3)!} 6V^e. \tag{A.2}$$

The derivation presented will follow the two dimensional form, but can easily be expanded to three dimensions. The key is to realize differential area ($dx\ dy$) in Equation A.1 can be illustrated by the parallelogram in Figure A.1. Let s_i as defined in the figure be the distance from edge $i+1 \mod 3$ in a parallel direction with

edge i to the parallelogram coordinate. Then we define $\xi_i = s_i/h_i$. Notice that the differential widths of the edges of the parallelogram are h_1 $\mathtt{d}\xi_1$ and h_2 $\mathtt{d}\xi_2$. Thus the differential area is

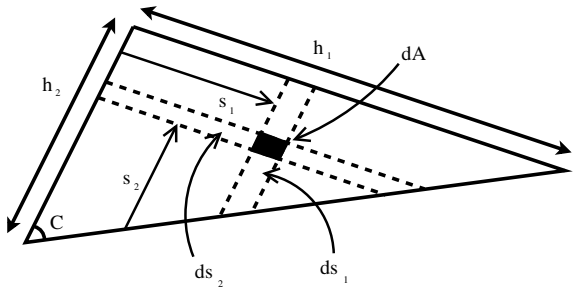


Figure A.1: Parallelogram differential area in a triangle.

$$dx\ dy = dA = (h_1\ d\xi_1)\ (h_2\ d\xi_2) \sin\alpha. \tag{A.3}$$

Since the formula for the area of a triangle is

$$\Delta^e = \frac{1}{2} h_1 h_2 \sin C \tag{A.4}$$

Equation A.3 simplifies to

$$dA = 2\Delta^e\ d\xi_1\ d\xi_2. \tag{A.5}$$

where ξ_i are the normalized area coordinates. Since $\xi_1 + \xi_2 + \xi_3 = 0$

$$\int_A \xi_1^l \xi_2^m \xi_3^n\ dA = 2\Delta^e \int_0^1 \left[\int_0^{1-\xi_1} \xi_1^l \xi^m (1 - \xi^l - \xi^m)\ d\xi_2 \right]\ d\xi_1. \tag{A.6}$$

Defining $t = \xi_2/(1-\xi)$ replaces the inner integral as

$$\int_A \xi_1^l \xi_2^m \xi_3^n \, dA = 2\Delta^e \int_0^1 \xi_1^l (1-\xi_1)^{m+n+1} \, d\xi_1 \int_0^1 t^m (1-t)^n \, dt \tag{A.7}$$

The integrals on the right hand side of the equation are a form of the beta function [1]

$$B(z,w) = \int_0^1 t^{z-1}(1-t)^{w-1} \, dt = \frac{\Gamma(z)\Gamma(w)}{\Gamma(z+w)} \tag{A.8}$$

where $\Gamma(x)$ is the gamma function, which satisfies $\Gamma(n+1) = n!$ for integers $n \geq 0$. Thus,

$$\begin{aligned} \iint_{\Omega^e} \left(\phi_i^e(\mathbf{r})\right)^l \left(\phi_j^e(\mathbf{r})\right)^m \left(\phi_k^e(\mathbf{r})\right)^n \, dx \, dy &= && \text{(A.9)} \\ &= 2\Delta^e \frac{\Gamma(l+1)\Gamma(m+1)\Gamma(n+1)}{\Gamma(l+m+n+3)} && \text{(A.10)} \\ &= \frac{l!\, m!\, n!}{(l+m+n+n+2)!} 2\Delta^e && \text{(A.11)} \end{aligned}$$

Q.E.D.

APPENDIX B

TABLE OF SYMBOLS AND FORMULAE

$p(\hat{\mathbf{s}}, \hat{\mathbf{s}}')$	the phase function of the angle between $\hat{\mathbf{s}}$ and $\hat{\mathbf{s}}'$.
g	the mean cosine of the scattering angle θ given by Equation B.5
μ	$\cos\theta = \hat{\mathbf{s}} \cdot \hat{\mathbf{s}}'$
ρ	particle density
μ_a	absorption cross section
μ_s	scattering cross section
μ_t	extinction cross section $(\mu_a + \mu_s)$
μ_{tr}	transport cross section $(\mu_s(1-g) + \mu_a)$
$I_d(\mathbf{r}, \hat{\mathbf{s}})$	diffuse intensity
$I_{ri}(\mathbf{r}, \hat{\mathbf{s}})$	reduced incident intensity given by Equation B.3
$F_d(\mathbf{r})$	diffuse flux vector, whose direction is given by a unit vector $\hat{\mathbf{s}}_\mathbf{f}$ defined in Equation B.7
$\varepsilon(\mathbf{r}, \hat{\mathbf{s}})$	source function
$\varepsilon_{ri}(\mathbf{r}, \hat{\mathbf{s}})$	source function due to reduced incident intensity see Equation B.4
$U_d(\mathbf{r})$	average diffuse intensity given by Equation B.2
$U_{ri}(\mathbf{r})$	average reduced intensity given by Equation B.1
β	$(\mu_{tr})^{-1}$
λ	$3\mu_a$
ν	$3\mu_s$
S	$\nu U_{ri}(\mathbf{r}) - \frac{3}{4\pi}\nabla \cdot \left[\beta \int_{4\pi} \varepsilon_{ri}(\mathbf{r}, \hat{\mathbf{s}})\hat{\mathbf{s}}\, d\omega\right]$

$$U_{ri}(\mathbf{r}) = \frac{1}{4\pi} \int_{4\pi} I_{ri}(\mathbf{r}, \hat{\mathbf{s}})\, d\omega \tag{B.1}$$

$$U_d(\mathbf{r}) = \frac{1}{4\pi} \int_{4\pi} I_d(\mathbf{r}, \hat{\mathbf{s}}) \tag{B.2}$$

$$\frac{dI_{ri}(\mathbf{r}, \hat{\mathbf{s}})}{ds} = -\mu_t I_{ri}(\mathbf{r}, \hat{\mathbf{s}}) \tag{B.3}$$

$$\varepsilon_{ri}(\mathbf{r},\hat{\mathbf{s}}) = \frac{\mu_t}{4\pi}\int_{4\pi} p(\hat{\mathbf{s}},\hat{\mathbf{s}}')I_{ri}(\mathbf{r},\hat{\mathbf{s}})\ d\omega' \tag{B.4}$$

$$g = (p(\hat{\mathbf{s}},\hat{\mathbf{s}}'))\,\mu\ d\omega\left(\int_{4\pi} p(\hat{\mathbf{s}},\hat{\mathbf{s}}')\ d\omega'\right) \tag{B.5}$$

$$\mathbf{Q_1}(\mathbf{r}) = \frac{\mu_t}{4\pi}\int_{4\pi}\left[\int_{4\pi} p(\hat{\mathbf{s}},\hat{\mathbf{s}}')\hat{\mathbf{s}}\ d\omega\right] I_{ri}(\mathbf{r},\hat{\mathbf{s}}')d\omega' \tag{B.6}$$

$$\mathbf{F}_d(\mathbf{r}) = \int_{4\pi} I_d(\mathbf{r},\hat{\mathbf{s}})\hat{\mathbf{s}}\ d\omega = F_d(\mathbf{r})\hat{\mathbf{s}}_\mathbf{f} \tag{B.7}$$

BIBLIOGRAPHY

[1] M. Abramowitz and I. Stegun, editors. *Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables*. National Bureau of Standards Applied Mathematics Series, 1974.

[2] M. Ainsworth and J. T. Oden. A posteriori *error estimation in finite element analysis*. John Wiley, New York, 2000.

[3] S. Arridge, H. Dehghani, M. Schweiger, and E. Okada. The finite element model for the propagation of light in scattering media: A direct method for domains with nonscattering regions. *Med. Phys.*, 27(1):252–264, January 2000.

[4] S. R. Arridge and J. C. Hebden. Optical imaging in medicine ii. modeling and reconstruction. *Med. Phys.*, 42(841), 1997.

[5] E. Aydin, C. de Oliveira, and A. Goddard. A comparison between transport and diffusion calculations using a finite element-spherical harmonics radiation transport method. *Med. Phys.*, 29(9), September 2002.

[6] G. Bal. Particle transport through scattering regions with clear layers and inclusions. *Journal of Computational Physics*, 180:659–685, 2002.

[7] J. F. Blinn. Light reflection functions for simulation of clouds and dusty surfaces. *Computer Graphics*, pages 21–29, July 1982.

[8] D. A. Boas, M. A. Franceschini, A. K. Dunn, and G. Strangman. *In Vivo Optical Imaging of Brain Function*, chapter Non-Invasive imaging of cerebral activation with diffuse optical tomography, pages 194–216. CRC Press, 2002.

[9] B. Brooksby, H. Dehghani, B. Pogue, and K. Paulsen. Near-infrared (nir) tomography breast image reconstruction with *A Priori* structural information from mri: Algorithm development for reconstructing heterogeneities. *IEEE Journal of Selected Topics in Quantum Electronics*, 9(2):199–209, March/April 2003.

[10] N. Carr, J. Hall, and J. Hart. Gpu algorithms for radiosity and subsurface scattering. In *HWWS '03: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 51–59, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[11] S. Chandrasekhar. *Radiative Transfer*. Dover Publications, Inc., 1960.

[12] W.-F. Cheong, S. Prahl, and A. Welch. A review of the optical properties of biological tissues. *IEEE Journal of Quantum Electronics*, 26(12):2166–2185, December 1990.

[13] G. R. Cowper. Gaussian quadrature formulas for triangles. *International Journal for Numerical Methods in Engineering*, 7(3):405–408, 1973.

[14] C. Dachsbacher and M. Stamminger. Translucent shadow maps. In *Proceedings of the 14th Eurographics workshop on Rendering*, pages 197–201. Eurographics Association, 2003.

[15] N. Davidson. *Sky Phenomena: A Guide to Naked Eye Observation of the Heavens*. FlorisBooks, Edinburgh, 1993.

[16] H. Dehgahani and D. Delpy. Near-infared spectroscopy of the adult head: effect of scattering and absorbing obstructions in the cerebrospinal fluid layer on light distribution in the tissue. *Applied Optics*, 39(25):4721–4729, September 2000.

[17] M. Eisenberg and L. Malvern. N finite element integration in natural coordinates. *International Journal for Numerical Methods*, 7:574–575, 1973.

[18] M. Firbank, S. Arridge, M. Schweiger, and D. Delpy. An investigation of light transport through scattering bodies with non-scattering regions. *Phys. Med. Biol.*, 41:767–783, 1996.

[19] M. Firbank, M. Hiraoka, M. Essenpreis, and D.T. Delpy. Measurement of the optical properties of the skull in the wavelength range 650-950nm. *Physics in Medicine and Biology*, 38(4):503–510, April 1993.

[20] J. Fishkin, O. Coquoz, E. Anderson, M. Brenner, and B. Tromberg. Frequency-domain photon migration measurements of normal and malignant tissue optical properties in a human subject. *Applied Optics*, 36(1):10–20, January 1997.

[21] M.A. Franceschini and D. A. Boas. Noninvasive measurement of neuronal activity with near-infrared optical imaging. *Neuroimage*, 21(1):372–386, 2004.

[22] H. Fuchs, Z. Kedem, and B. Naylor. On visible surface generation by a priori tree structures. In *SIGGRAPH '80 Proceedings*, volume 14, pages 124–133, 1980.

[23] D. Garber. *Descartes' Metaphysical Physics*. University of Chicago Press, Chicago, 1992.

[24] C. Goral, K. Torrance, and D. Greenberg. Modeling the interaction of light between diffuse surfaces. In *SIGGRAPH '84 Proceedings*, volume 18, pages 213–222, 1984.

[25] G. Gratton, P.M. Corballis, E. Cho, M. Fabiani, and D.C. Hood. Shades of gray matter: noninvasive optical images of human brain responses during visual stimulation. *Psychophysiology*, 32(5):505–509, September 1995.

[26] D. A. Hall. *The Ageing of Connective Tissue*. Academic Press, London, 1976.

[27] X. Hao, T. Baby, and A. Varshney. Interactive subsurface scattering for translucent meshes. In *Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 75–82. ACM Press, 2003.

[28] P. Heckbert. Survey of texture mapping. *IEEE Computer Graphics and Applications*, 6(11):56–67, 1986.

[29] A. Hidenobu, E. Mariko, and Y Yukio. Depth profile of diffuse reflectance near-infrared spectroscopy for measurement of water content in skin. *Skin Research and Technology*, 11(1):27–35, 2005.

[30] Kitware Inc. VolView. http://www.kitware.com/products/volview.html.

[31] A. Ishimaru. *Wave Propagation and Scattering in Random Media*. Academic Press, New York, 1978.

[32] S. L. Jacques, C. A. Alter, and S. A. Prahl. Angular dependence of hene laser light scattering by human dermis. *Lasers Life Science*, 1:309–333, 1987.

[33] H. W. Jensen. *Realistic Image Synthesis using Photon Mapping*. AK Peters, 2001.

[34] H. W. Jensen, S. R. Marschner, M. Levoy, and P. Hanrahan. A practical model for subsurface light transport. In *Proceedings of the 28th annual conference on computer graphics and interactive techniques*, pages 511–518. ACM Press, 2001.

[35] V. John. A numerical study of *a posteriori* error estimators for convection-diffusion equations. *Comput. Methods Appl. Mech. Engrg.*, 190:757–781, 2000.

[36] J. T. Kajiya. The rendering equation. In *SIGGRAPH '86*, pages 143–150. ACM Press, 1986.

[37] K. Kalantar-Zadeh, E. Dunne, K. Nixon, K. Kahn, G.H. Lee, M. Kleiner, and F.C. Luft. Near infra-red interactance for nutritional assessment of dialysis patients. *Nephrol Dial Transplant*, 14(1):169–175, January 1999.

[38] G. Karypis. Metis: Family of multilevel partitioning algorithms. http://www-users.cs.umn.edu/ karypis/metis/.

[39] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20(1):359–392, 1998.

[40] J. Kniss, S. Premoze, C. Hansen, and D. Ebert. Interactive translucent volume rendering and procedural modeling. In *Proceedings of the conference on Visualization '02*, pages 109–116. IEEE Computer Society, 2002.

[41] J. Koenderink and S. Pont. The secret of velvety skin. *Machine Vision and Applications*, 14(4):260–268, September 2003.

[42] J. Lee, S. Kim, and Y. Kim. Modeling of diffuse-diffuse photon coupling via a nonscattering region: a comparative study. *Applied Optics*, 43(18), June 2004.

[43] H. Lensch, M.Goesele, P. Bekaert, J. Kautz, M. Magnor, J. Lang, and H.-P. Seidel. Interactive rendering of translucent objects. In *Proceedings of Pacific Graphics '02*, pages 214–224, October 2002.

[44] R. Marks and S.P. Barton. The significance of the size and shape of corneocytes. In R. Marks and G. Plewig, editors, *Stratum Corneum*. Springer Verlag, Berlin, 1983.

[45] G. Marquez, L. Wang, S. P. Lin, J. Schwartz, and S. Thomsen. Anisotropy in the absorption and scattering spectra of chicken breast tissue. *Applied Optics*, 37(4):798–804, 1998.

[46] S. Marschner, S. Westin, A. Arbree, and J. Moon. Measuring and modeling the appearance of finished wood. *ACM Transactions on Graphics*, 24(3):727–734, 2005.

[47] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

[48] T. Mertens, J. Kautz, P. Bekaert, F. V. Reeth, and H.-P. Seidel. Efficient rendering of local subsurface scattering. In *11th Pacific Conference on Computer Graphics and Applications (PG'03)*, pages 51–58, October 2003.

[49] T. Mertens, J. Kautz, P. Bekaert, H.P. Seidelz, and F. Van Reeth. Interactive rendering of translucent deformable objects. In *Proceedings of the 14th Eurographics workshop on Rendering*, pages 130–140. Eurographics Association, 2003.

[50] N. Metropolis and S. Ulam. The monte carlo method. *Journal of the American Statistical Association*, 44:335, 1949.

[51] H. Ohtsubo and M. Kitamura. Element by element *A Posteriori* error estimation and improvement of stress solutions for two-dimensional elastic problems. *International Journal for Numerical Methods in Engineering*, 29:223–244, 1990.

[52] D. W. Pepper and J. C. Heinrich. *The Finite Element Method: Basic Concept and Applications*. Hemisphere Publishing Corporation, 1992.

[53] K. Perlin. An image synthesizer. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 287–296, New York, NY, USA, 1985. ACM Press.

[54] S. Pinker. *How the Mind Works*. W. W. Norton & Company, Inc., 1997.

[55] K. H. Potter. *Indian Metaphysics and Epistemology*. Princeton University Press, Princeton, 1977.

[56] S. A. Prahl. *Light Transport in Tissue*. PhD thesis, University of Texas at Austin, 1988.

[57] T. J. Purcell, C. Donner, M. Cammarano, H. W. Jensen, and P. Hanrahan. Photon mapping on programmable graphics hardware. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 41–50. Eurographics Association, 2003.

[58] K. Sakatani, S. Chen, W. Lichty, H. Zuo, and Y.P. Wang. Cerebral blood oxygenation changes induced by auditory stimulation in newborn infants measured by near infrared spectroscopy. *Early Human Development*, 55(3):229–236, July 1999.

[59] M. Schweiger, S. Arridge, M. Hirako, and D. Delpy. The finite element method for the propagation of light in scattering media: Boundary and source conditions. *Med. Phys.*, 22(11):1779–1792, November 1995.

[60] R. Sharp and R. Machiraju. A simplified model for inhomogeneous subsurface scattering. In *Volume Graphics*, pages 63–72. EuroGraphics/IEEE Computer Society VGTC, 2005.

[61] R. Sharp and R. Machiraju. Accelerating subsurface scattering using cholesky factorization. *The Visual Computer*, pages 1–9, 2006.

[62] P. Shirley and A. Tuchman. A polygonal approximation to direct scalar volume rendering. In *VVS '90: Proceedings of the 1990 workshop on Volume visualization*, pages 63–70, New York, NY, USA, 1990. ACM Press.

[63] C.R. Simpson, M. Kohl, M. Essenpreis, and M. Cope. Near infrared optical properties of *ex-vivo* human skin and subcutaneous tissues measured using the monte carlo inversion technique. *Physics in Medicine and Biology*, 43:2465–2478, 1998.

[64] P.-P. Sloan, J. Kautz, and J. Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH*, July 2002.

[65] J. Stam. Multiple scattering as a diffusion process. In *Proceedings of the 6th Eurographics Workshop on Rendering*, pages 51–58, Dublin, Ireland, 1995.

[66] C.Y. Tan, B. Statham, R. Marks, and P.A. Payne. Skin thickness measurement by pulsed ultrasound: its reproducibility, validation and variability. *The British Journal of Dermatology*, 106(6):657–667, June 1982.

[67] S. Toledo, D. Chen, and V. Rotkin. Taucs: A library of sparse linear solvers. http://www.tau.ac.il/ stoledo/taucs/.

[68] J.A. Viator, B. Choi, G.M. Peavy, S. Kimel, and J.S Nelson. Spectra from 2.5-15 microm of tissue phantom materials, optical clearing agents and ex vivo human skin: implications for depth profiling of human skin. *Phys. Med. Biol.*, 48(2):N15–24, January 2003.

[69] J. P. Webb and S. McFee. Nested tetrahedral finite elements for h-adaption. *IEEE Transactions on Magnetics*, 35(3):1338–1341, May 1999.

[70] Eric W. Weisstein. Divergence theorem. http://mathworld.wolfram.com/DivergenceTheorem.html. From MathWorld–A Wolfram Web Resource.

[71] X. Xu, W. Zhu, V. Padival, M. Xia, X. Cheng, R. Bush, L. Christenson, T. Chan, T. Doherty, and A. Iatridis. Validation of nirs in measuring tissue hemoglobin concentration and oxygen saturation using benchtop and isolated limb model. In *Proceedings of SPIE*, volume 4955, pages 369–378, 2003.

[72] G. Yoon. *Absorption and Scattering of Laser Light in Biological Media – Mathematical Modeling and Methods for Determining Optical Properties*. PhD thesis, University of Texas at Austin, 1988.